Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Panajis Rantala

MitMUI:
Man in the Middle approach for User Interface Prototyping

Master's Thesis
Espoo, May 26, 2019

Supervisors: Professor Marko Nieminen, Aalto University
Advisor: Mika P. Nieminen D.Sc. (Tech.)

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author:	Panajis Rantala		
Title:	MitMUI: Man in the Middle approach for User Interface Prototyping		
Date:	May 26, 2019	**Pages:**	x +54 +15
Major:	Computer Science	**Code:**	SCI3042
Supervisors:	Professor Marko Nieminen		
Advisor:	Mika P. Nieminen D.Sc. (Tech.)		

The prototyping of products, ideas, mobile applications and web applications has been done extensively, and there exists a plethora of tools for this. These tools haven't seen big changes lately. Additionally, the amount of different front-end frameworks has risen substantially. This means that web development isn't as simple as it once was. Many different frameworks have their individual syntaxes, paradigms and workflows. This makes it harder for new developers to get started on existing projects, as familiarization might take longer than expected.

At the same time, more and more further development is being done by companies. This means that a company has to fix, enhance and develop an already existing service created by others. The design changes and prototyping might be very labour intensive, as the new company must usually build the new designs from scratch.

This thesis attempts to study different prototyping methods and tools, and also presents a new approach to prototyping. This is based on an iFrame element, which is injected to the bottom of the document object model. This enables the designers and developers to use an already existing site as the basis for further development. This will save resources and reduce the workload of designers and developers.

MitMUI, Man in the Middle User Interface will be conceptualized based on already existing prototyping tools. Additionally, further information will be researched from existing literature and restrictions discovered. The concept will be evaluated by executing structured interviews, which will also discover the general benefits and drawbacks to prototyping tools. The results show, that there is a need for this type of an approach. MitMUI receives much praise, and the consensus is that it would be useful.

Keywords:	web application, iFrame, prototyping, user interface,
Language:	English

Aalto-yliopisto
Perustieteiden korkeakoulu
Tieto-, tietoliikenne- ja informaatiotekniikan maisteriohjelma

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Panajis Rantala		
Työn nimi:	MitMUI: Man in the Middle approach for User Interface Prototyping		
Päiväys:	26. toukokuuta 2019	**Sivumäärä:**	x +54 +15
Pääaine:	Tietotekniikka	**Koodi:**	SCI3042
Valvojat:	Professori Marko Nieminen		
Ohjaaja:	Tekniikan tohtori Mika P. Nieminen		

Tuotteiden, ideoiden, mobiilisovellusten ja internet-sivujen prototypointia on harrastettu jo pitkään ja siihen on olemassa monia työkaluja. Nämä työkalut eivät kuitenkaan ole juuri kehittyneet viimeaikoina. Erilaisia työkaluja on paljon, kuten myös käyttötapoja. Lisäksi erilaisten front-end kehittämiskehysten määrä on kasvanut. Tämä tarkoittaa sitä, että web-kehittäminen ei ole niin yksiselitteistä, kuin se aiemmin oli. Monilla eri kehityskehyksillä on omat syntaksinsa, paradigmansa sekä työnkulkunsa. Tämä vaikeuttaa monien uusien kehittäjien työtä, sillä projektin sisäistämiseen saattaa kulua enemmän aikaa kuin odotettiin.

Samaan aikaan yritykset tekevät enenevissä määrin jatkokehitystä, jolloin yrityksen tehtävänä on parantaa, korjailla ja kehittää edelleen muiden aluperin tekemää, jo olemassaolevaa sivustoa. Tällöin uusien käyttöliittymien suunnitteluun ja prototypointiin kuluu myös paljon resursseja, sillä uuden yhtiön täytyy rakentaa ne usein tyhjästä.

Tämä diplomityö pyrkii tutkimaan erilaisia prototypointimenetelmiä ja työkaluja, ja esittelee uudenlaisen lähestymistavan prototypoida. Tämä perustuu iFrame elementtiin, joka sisällytetään alimmaiseksi sivuston dokumenttioliomalliin. Se mahdollistaa jo olemassaolevan sivuston käyttämisen jatkokehityksen pohjana. Tämä nopeuttaa sekä suunnittelijoiden, että kehittäjien työtä.

MitMUI, Man in the Middle User Interface eli ”Mies Välissä - käyttöliittymä”konseptoidaan jo olemassaolevia prototypointityökaluja hyväksikäyttäen. Lisäksi kirjallisuudesta haetaan tukea ja rajoituksia tälle lähestymistavalle. Konseptia testataan suorittamalla haastatteluja, joissa myös pyritään selvittämään millaisia hyviä ja huonoja puolia jo olemassaolevissa prototypointityökaluissa on. Tutkimustuloksista nähdään, että tällaiselle menetelmälle olisi käyttöä, ja että se vaikuttaa toimivalta.

Asiasanat:	verkkosovellus, iFrame, prototypointi, käyttöliittymä
Kieli:	Englanti

Acknowledgements

I wish to express my most sincerest gratitude to the faculty of Aalto University, for making my employment there a very pleasant one. A special thank you for my advisor, Mika P. Nieminen, for putting up with my constant barrage of questions, and for his stern but fatherly guidance. Also a big thank you for the thesis supervisor Professor Marko Nieminen for all his domain knowledge and help during this ordeal.

Thank you mom for never doubting me, and always reminding me to rest rather than overwork. All your reassurance really helped me during all the years of my studies and life.

Finally, a colossal thank you for my fiancée for constantly reminding me to keep working at this thesis, and all the endless incentives, help and encouragement she gave me.

Espoo, May 26, 2019

Panajis Rantala

Abbreviations and Acronyms

SOP	Same-Origin Policy
XSS	Cross Site Scripting
JS	Javascript
HTML	Hypertext Markup Language
CSS	Cascading Style Sheet
UI	User Interface
CORS	Cross-Origin Resource Sharing
SDLC	Software Development Life Cycle
NLU	Natural Language Understanding
CMC	Computer Mediated Communication
HCI	Human-Computer Interaction
API	Application Programming Interface

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As long as products, be it physical, electronic or other have been manufactured, prototypes were needed before that. In the beginning, they used to be just quick scribbles on a paper but as technology evolved, so did the prototyping methods. Prototype as a word, and its derivatives can mean different things e.g. prototyping new concepts, producing a prototype of a concept, or writing prototype programs (Ulrich, 2003). As in Ulrich (2003), a prototype is defined in this thesis as "an approximation of the product along one or more dimensions of interest.".

Prototyping, as we know it, started off as a buzzword around 1980's. However, in the timespan of 10 years, it evolved to be something companies actually used to document development process. This was somewhat hastened by the need for requirement analysis. Software could only be tested after it was written, which in turn was expensive. Hence the need for prototypes arose. (Budde and Zullighoven, 1990)

A plethora of different prototyping methods exist with different application domains. Depending on the fidelity requisite of the prototype, further approaches may be taken. For a long time however, all prototypes have been very static. Even with the advent of interactive prototyping tools which allow user input and animations, none provide the possibility to design the prototype against live production data.

This thesis attempts to implement a new form of prototyping, which would allow "live" prototyping against virtually any website, regardless of the underlying technology used. MitMUI, Man in the Middle User Interface uses an iFrame to show an existing website and lays elements on top of it and its own separate "shadow" Document Object Model (DOM).

This enables designers to quickly experiment with changes or small additions without prior knowledge of the underlying site structure, pedantic hierarchy or convoluted events and triggers. Furthermore, the final proto-

type produced by the approach, should be easily implementable to the site. Since the prototype is created along with the designing, this is expected to hasten the development spanning from designing to implementation.

1.1 Goals of the thesis

1.1.1 Research goals

The research goal of this thesis is to produce a new viable prototyping approach to allow quick prototyping against live website. The technologies used in the live data should not matter to the end user. For this reason it is important to study the applicability of an iFrame element for a viable prototyping method for accessing live data. Furthermore this thesis will test and analyze some modern prototyping tools, to act as comparison and provide background information about them.

1.1.2 Research questions

The research questions of the thesis focus mainly on the applicability and validity of the proposed solution. This thesis attempts to find solutions to the following research questions:

- *RQ1:* What kinds of tools are already available for prototyping web applications?

  Do any of them allow live data to be used?

  Are there any tools that would utilize iFrame?

- *RQ2:* What makes a prototyping tool practical?

  Is the use of an iFrame applicable for a prototyping tool?

  What kind of restrictions does using an iFrame pose?

- *RQ3:* How does the MitMUI approach perform compared to other methods?

Answers to research question 1 can be found by studying the relative literature, and concrete testing of different prototyping tools. Research question 2 can also benefit from literature, but interviews with relevant designers and developers would provide lusher information for research questions 2 and 3. To further extract useful information, content analysis will be used (Lindlof and Taylor, 2017). These methods will be further elaborated in section 4.1.

1.1.3 Research scope and constraints

Within the time frame of this thesis, there is not a possibility to produce a product of commercial quality. This thesis aims to answer the aforementioned research questions which help the proper implementation in the future. Through a semi-structured interview, an answer will be acquired whether this type of prototyping method is a viable one compared to others currently available tools and methods. Chapter 2 will delve deeper into already existing prototyping tools and methods to provide supporting and relevant information in the field of prototyping.

1.2 Environment

The idea for this thesis arose out of necessity. Aalto University was a research partner in developing Eike, an intelligent chatbot for answering questions from immigrants and refugees in Germany. The project was initiated by Technische Universität Berlin (TU Berlin) with T-System Multimedia Solutions being the customer. Aalto University was tasked with developing the front-end and performing user experience and usability testing.

The Eike project was developed with Flask web microframework. This allowed for a simple and quick implementation (Ronacher, 2019). For the machine learning, Rasa CORE and NLU were used with some 2000 lines of training data. The original data during the development phase was manually extracted by Aalto University from the Handbook Germany website. In other revisions the data was provided by TU Berlin, and it was crowdsourced from real immigrants for maximum accuracy.

The frontend included the basics of web development, HTML, jQuery, less and a simple animation libary animate.css. The frontend was developed iteratively in a joint cooperation with a designer from Aalto University.

1.2.1 Background

While Handbook Germany, the website aimed towards helping immigrants and refugees to integrate better into Germany already features a plethora of useful information, it was discovered that the customer base had trouble finding relevant information. This might have been due to lacking information retrieval skills or just a simple language barrier issue.

Nonetheless, a new method for reaching the users was needed. As chatbots seem to be on the rise, (see figure 1.1 from Google Trends (2019)) such a system was decided to be added to the site. Studies have shown, that with

the advent of natural language understanding (NLU) methods and computer mediated communication (CMC) the usability of such services have grown to the point of chatbots passing Turing tests (Hill et al., 2015).

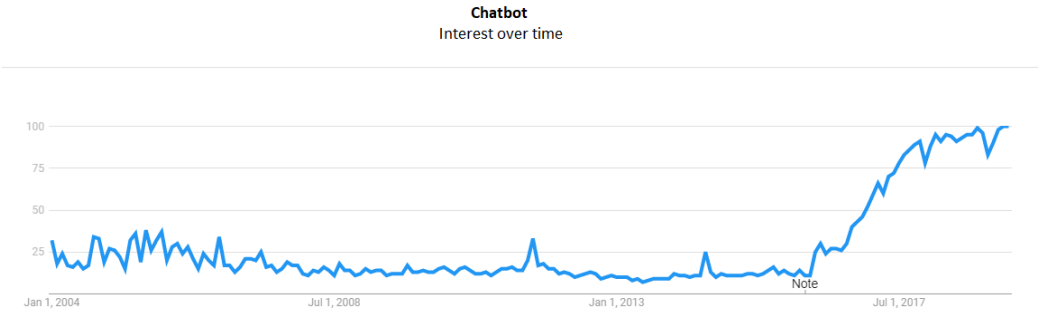


Figure 1.1: Google Trends chart for chatbot. Note: Google's improved data collection system was implemented 1.1.2016

Aalto University as an actor in this project decided to use co-design as a paradigm to better understand the requirements for the chatbot's end users i.e. immigrants and refugees. Co-design has been shown to be beneficial especially in cases of asymmetrical relations between the developers and end users (Hussain et al., 2012). Therefore, in the course of this project, multiple usability testing workshops and brainstorming sessions were held. These are, however, not the focal point of this thesis.

1.2.2 Motivation

When the assignment was given to Aalto University to develop the frontend for the chatbot, no access to source code was provided. After enquiring, it was stated that the target domain's source code would not be made available for Aalto University. This made it hard to develop a seamlessly integratable chatbot for the site.

As a quick fix, iFrame was used to display the site. This made it easy to design and develop the chatbot "on top of" the target site. This approach was never meant to be the final version, but as the development progressed it proved to be a very viable concept. This sparked the interest in determining whether this approach would be a viable means of prototyping and developing, especially against live production data.

After a cursory search, a multitude of prototyping tools presented themselves. However, there was no clear comparisons or differences in functionalities available. As a developer with personal needs for prototyping, this

thesis created the perfect opportunity to research various prototyping methods and tools, while developing a completely new approach. This approach might prove to be especially useful for developers, who wish to quickly 'slap together' designs and rapidly iterate on their projects.

1.2.3 Partners and Roles

Aalto University
Aalto University was tasked with performing user research and designing and implementing the front-end for the Eike Chatbot. This was done by conducting a number of workshops, questionnaires and other usability evaluation methods. The results from these were analyzed, and based on them, a visual prototype was created by a designer. Finally, the frontend for Eike was developed by the author.

T-System Multimedia Solutions (Part of Deutsche Telekom)
T-Systems MMS hosts the target domain Handbook Germany for Eike chatbot. Handbook Germany is a web portal aimed for immigrants and refugees coming to German, and has information about different aspects of German culture as well as practical information e.g. getting an education or applying for different benefits (T-Systems Multimedia Solutions, 2019). In addition to offering their site as the pilot platform, T-systems MMS is also responsible for the commercial launch of the whole Handbook Germany's Eike project.

Technische Universität Berlin
TU Berlin first and foremost developes crowdbased solutions (Berlin, 2019). Additionally they are also in the lead role of the project. In later stages, Eike is to be augmented with crowdsourced translators and communicators, if the AI fails. TU Berlin also provided the training data for the chatbot via crowdsourcing.

Deutsches Forschungzentrum für Künstliche Intelligenz (DFKI)
DFKI is the German Research Center for Artificial Intelligence (DFKI, 2019). They were responsible of the backend and machine learning aspects of Eike chatbot. From a development standpoint, Eike was created by Aalto University and DFKI.

1.3 Research process & approach

The idea of design science is to build artefacts or theories which are relevant in some form for the environment. Previous knowledge will be applied rigorously in the development of new artefacts. These are then assessed using a toolbox of applicable approaches e.g. case studies or analytical evaluation,

after which the artefact will be refined further. At the end of this iterative loop, the newly developed artefact should answer the need for the environment, while the newly accrued information serves as addition to the existing knowledge base of the subject. (Hevner et al., 2004)

The requirement for devising an iFrame based prototyping environment presented itself as a business need. It was critical to have the target domain visible, to properly start developing EIKE chatbot. As the process matured, a question arose. Had this approach been previously used for prototyping, and if not, why? After a cursory check revealed no similar previous work, a thesis was formed to research this idea further.

Chapter 2 delves into prototyping as a science, and showcases some time tested methods and modern tools. These will be compared to formulate internal requirements and functionality needed for the MitMUI approach. This information will be put to use in chapter 3, where the requirements, functionality, constraints and restrictions will be assessed. Furthermore it presents a synthesis about what the MitMUI will be, and how it would be implemented.

To fully apply the design science approach, the prototyped artefact will be evaluated using semi-structured interviews aimed at developers and designers. These answers will be analyzed using content analysis and Instant Data Analysis. The results for these will be shown in chapter 4. Finally, all non-applicable refinements and overall assessment of the conceived MitMUI approach will be evaluated and further development ideas will be presented.

Hevner et al. (2004) proposed a conceptual framework shown in figure 1.2 to help understand, execute, and evaluate information science research. For this thesis, the environment comprises of the entirety of organizations and people in charge of creating Eike. Furthermore, the target site's infrastructure and requirements are also included in the environment. The knowledge base on the other hand consists of material presented in chapter 2, including the data analysis techniques and validation criteria used. The information research is ultimately this whole thesis, as it attempts to synthesize a new concept for prototyping, using the existing knowledge and restrictions, tailored for the environment.

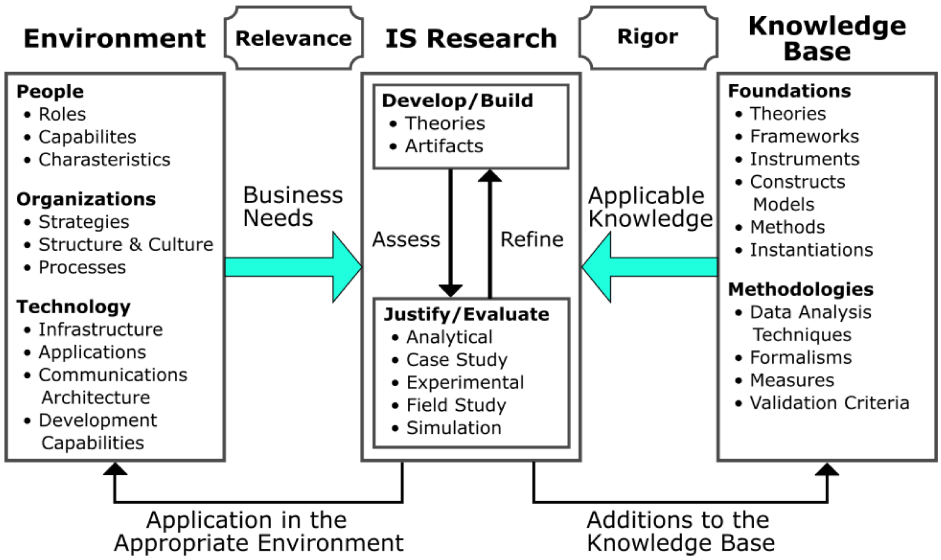


Figure 1.2: Hevner et al. (2004)'s chart showing the interwovenness of environment, information science research and knowledge base

Chapter 2

Background

This chapter offers a small cross-section of the needed technologies, some history and cutting edge methods used in the field of web applications and prototyping.

2.1 Prototyping

Prototyping is an important phase of any development process. Prototyping refers to the to the act of creating interim products which focus on some aspects of the product that need testing. Prototyping allows to quickly find flaws and especially shortcomings that would not have been obvious before seeing the finished product (Snyder, 2003). It also provides a quick way of getting feedback without investing excessive resources, be it time or money (Overmyer, 1991). Furthermore, prototyping allows better and increased user involvement as it invites users to actually experiment with something concrete (Crinnion, 1991).

Crinnion (1991) also states, that prototypes are especially useful in HCI-rich applications, where user interaction is abundant. This makes prototyping an extremely useful way to quickly gain insight into a web application development process in its early stages. Prototyping can however, especially if used in excess, lead to problems. Possible problems can arise from developer attachment, user confusion with prototype and final product, existing technology mismatch, lack of resources or experience etc (Rome, 1992).

Classically, prototypes are categorized in a polarizing manner, low-fidelity or high-fidelity prototypes (Walker et al., 2002). These refer to the level of completeness and effort required to build the prototype. An example of a low-fidelity prototype would be a paper prototype of a user interface with little interaction. A high-fidelity prototype could be a modern, interactive

web application, where most of the graphics and most used functionalities would be present.

McCurdy et al. (2006) however discusses about the problem called *fidelity barrier*. This refers to the multimodality and multidimensionality differences between different prototypes which would be considered either low- or high-fidelity. McCurdy et al. (2006) also presents a new 5-dimensional scheme for categorizing prototypes, which is referred to as mixed-fidelity. This allows for, for example, a high *Level of Visual Refinement* prototype with low *Depth of Functionality*. This grants a much wider spectrum for prototype categorizations.

2.1.1 Prototyping methods

There exists a variety of different approaches to prototyping, most differing by the application field. Prototypes are often used to answer questions like "Will it work?" and "How well does it meet the customer needs?" (Ulrich, 2003). This makes them perfect for the quick-paced and often agile environment of (web) software development. This subsection discusses methods mostly pertaining to software development life cycle (SDLC).

Paper prototyping refers to the most basic way of designing a prototype which only calls for sketches of the user interface or product. Paper prototyping is usually the fastest approach for quickly wireframing any ideas. While elemental, a higher fidelity can be achieved. A higher-fidelity prototype would have multiple views, which would be changed by the designer in reaction to user interaction. This way the user experience can also be tested with only pen and paper (Snyder, 2003). Paper prototyping is theoretically a subset of throwaway prototyping, but with its iconic nature, it is usually referred to as its own method.

Throwaway prototyping or more recently *rapid prototyping* is an approach, where emphasis is on quick-and-dirty, usually pen and paper prototypes. These are meant to serve more as ideation, and not to be included in the final prototype. Rapid prototyping has been succesfully used for many years. According to Gordon and Bieman (1995), out of 39 cases where rapid prototyping was used that they studied, 33 of them were successes.

This is in contrast to *Evolutionary prototyping*, where usually one prototype (and supporting architecture) is incrementally refined, until it becomes the basis for the final product. The benefit of this method lies in the acknowledgement, that not all requirements are known in the beginning of the development process (Davis, 1992). This way, more functionality and complexity can be added to the system before it becomes the final product. However, the method has been criticised of creating "design baggage" (Gor-

don and Bieman, 1995). This refers to inefficient code quickly created for the initial prototype, which could remain unchanged throughout the iterative process. This in turn might lead to non-optimal performance once the prototype "evolves" into a production environment.

Incremental prototyping follows a somewhat similar development manner as evolutionary prototyping. The difference being, that the final product is built from independent, separate prototypes which are merged in the end. Martin (2002) states, that this form of prototyping has been the basis for the now widely adopted, modern agile software development methods. This method has been said to reduce the initial stage prototyping, saving resources in the long run (Elverum and Welo, 2015).

Extreme prototyping is usually used especially in web development. This method shares similarities with many other methods. The development process is divided into three phases, the first being website hierarchy. In the beginning, a wireframe is built with HTML of all the necessary elements. The second phase entails implementing functionality with simulated services. In this stage, a website might appear as ready but it doesn't possess any off-site functionality. Finally, in the third phase, the services are implemented. (Ali, 2017)

2.1.2 Tools

While there exists a multitude of prototyping tools available both as web applications or native software, none of these employ a similar workflow to MitMUI. Most of the tools available are wireframing tools at heart. While they offer many advantages to traditional pen and paper prototyping, they're still inherently separate from any actual product in live production. As of writing this thesis, none of the commercially available tools offer a possibility of using a live site as a starting point for prototyping.

The thesis presents some of the modern prototyping tools next. While many of the following tools are inherently different, it is still worthwhile to study which features are required and useful for a prototyping tool.

2.1.2.1 PROTO.IO

Proto.io (Proto.io, 2019) is a relatively old application prototyping and wireframing tool made available as of 2012. Proto.io is designed for creating mobile application prototypes, and as such, isn't usable for web application prototyping. It does, however, offer interesting insights into the functionalities and layout of prototyping tools generally.

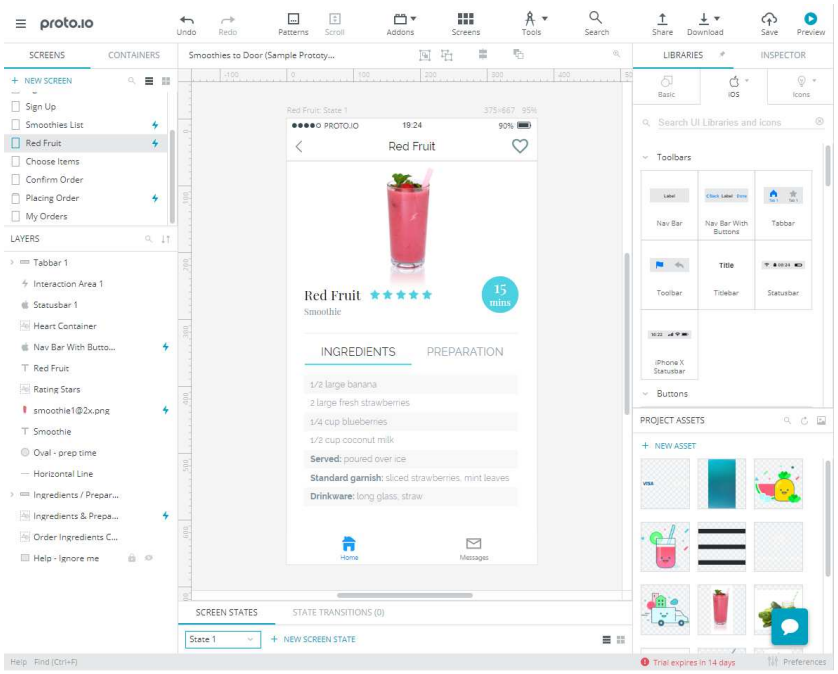


Figure 2.1: Basic view of Proto.io displaying the screens menu on the left, current screen in the middle, element libraries on the right

Figure 2.1 shows the main layout of the browser based tool. The design is very clean, and a quick tutorial in the beginning instructs about the main sections of Proto.io. All created screens (i.e. views) of the application prototype are shown on the left, and can be navigated quickly. On the right-hand side, visual elements such as icons or buttons for different operating systems can be browsed. The interface uses a simple drag and drop approach for creating the actual screen, which can be seen in the middle.

Figure 2.2 shows the normally hidden interactions pane on the right. Here user actions can be mapped to a plethora of prototype reactions e.g. changing screens, moving or hiding elements, or interacting with the prototypes inner variables. Multiple online reviews of the tool also commend its animation tweaking capabilities (G2crowd, 2019).

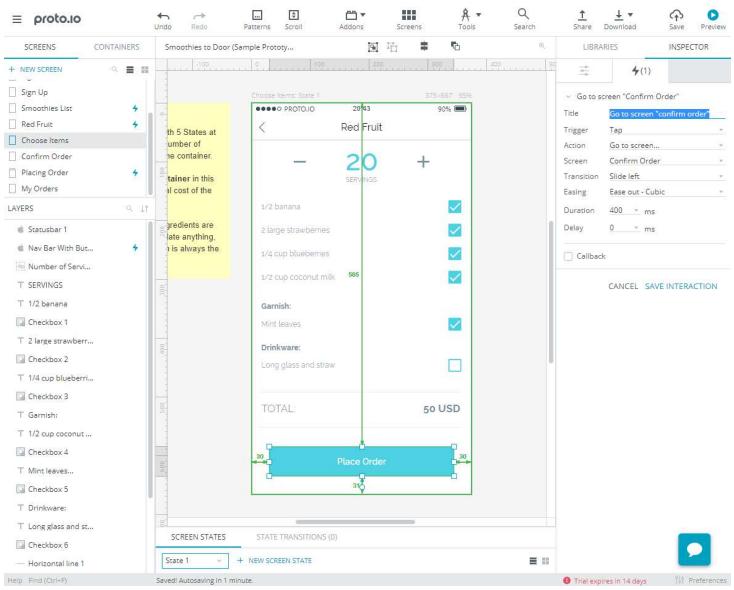


Figure 2.2: Main view of Proto.io displaying the interaction manager on the right. The transition and easing function drop-downs are noteworthy.

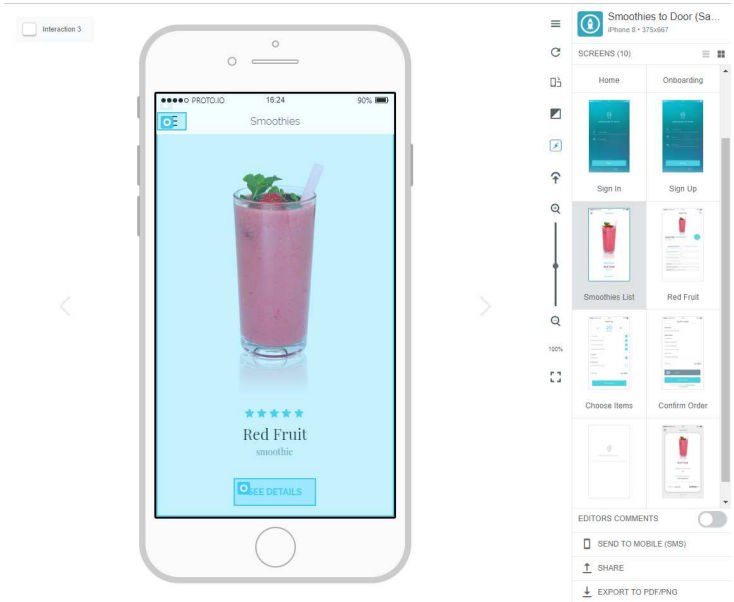


Figure 2.3: Interactive Preview view of Proto.io, showing all the screens in the prototype on the right menu

Figure 2.3 displays the inbuilt preview tool for the created prototype. The application can be used as if it were a real mobile application. All normal ways of interacting are supported, such as clicking and swiping the screen. Shortcuts for different screens are also offered on the right.

While Proto.io offers good tools for sharing and collaborating in the prototype design process and a great deal of functionality, it is only a wireframing tool. Proto.io offers export as different image- or pdf-files, and also as a static HTML page, but does not produce any code for a mobile device. It also cannot be used offline, and requires a proper internet connection or it might suffer from lag issues (G2crowd, 2019).

2.1.2.2 Axure

Axure is a powerful prototyping and wireframing tool with a plethora of other functionality built in. It has also been in development for a long time, and boasts being used in 87% of fortune 100 companies (Axure, 2019). In addition to prototyping interfaces for desktop and web applications, it also offers dynamic content, conditional flow and basic mathematical functionalities. It does not, however support live data from other services, like MitMUI.

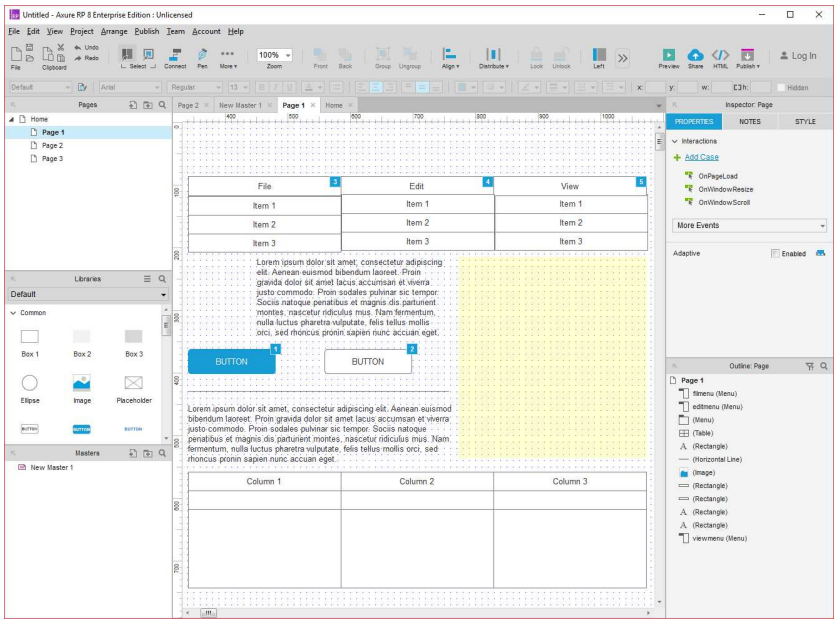


Figure 2.4: Basic view of Axure RP program displaying the pages menu and libraries on the left, current page in the middle, element inspector on the right

Axure focuses on collaboration. It provides tools for teams to collectively go through and leave remarks on the prototype, versioning support and even hosting of the prototype online. These make it very useful for large companies (Axure, 2019). Unlike Proto.io, which was solely for designing mobile applications, Axure can be used for mobile, website and naturally for all around prototyping e.g. for native programs.

Axure has a drag and drop editor, which allows the user to create simple mockups very swiftly. A very detailed rendition or a high-fidelity prototype might prove to be a bit problematic, with its ready made elements though. Events can be attached to all elements to provide further functionality, such as taking the user to another view. Figure 2.4 shows Axure's user interface, which is very simplistic, and even a bit old fashioned, with its multiple panes on both sides of the canvas. However, these seem to work very well and provide often used tools for quick usage.

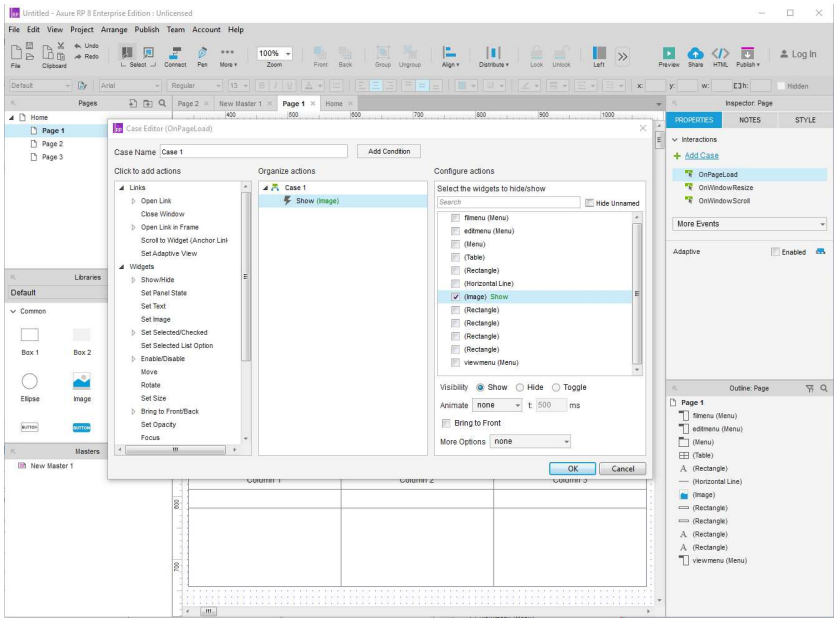


Figure 2.5: Axure RP interaction tools

Figure 2.5 shows Axure's interaction tools. One can attach a wide variety of actions to some of the usual triggers for example on click. Animations are handled elsewhere, which does add some complexity compared to Proto.io. Figure 2.6 shows the preview of the project, which automatically opens in a browser. As before, a pane for different pages is shown on the left for quick access, and the prototype can be interacted with.

Axure can also export the prototype as HTML. This does not, however, work without Axure's own libraries. The produced HTML for the simple interface shown in figure 2.6 can be seen in appendix D.

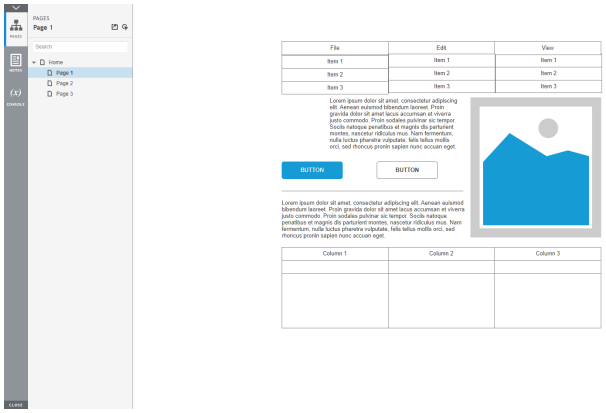


Figure 2.6: Axure RP interaction view. Available views are on the right.

2.1.2.3 Adobe XD

Adobe XD (Creative Cloud) is Adobe's take on prototyping tool. It allows the creation of high-fidelity, especially graphical prototypes (Adobe, 2019). The workflow is similar to Axure, as it offers dynamic content and some functionality, but offers less freedom for animations and prototype preview. Adobe XD can be used to create mobile, native or website wireframes and prototypes.

Figure 2.7 shows the normal layout of the Adobe XD CC native application. The view is much less cluttered compared to the previous tools. A simple toolbar can be seen on the left side, with assets and layers menus hidden under a button on the bottom. Right side houses an inspection panel, which changes its contents based on context.

Adobe XD also differs from previous tools in the way it handles interactions. Wireframing and actual design is done inside the Design mode, and adding interactions is done in a separate panel called Prototype. Figure 2.8 shows the Prototype view, where all the artboards are visible, and interactions can be added between elements. Upon creating a link between two elements, a minimal menu is shown to elaborate on the interaction and possible animations. The animation choices, however, are much less limited that in previous tools.

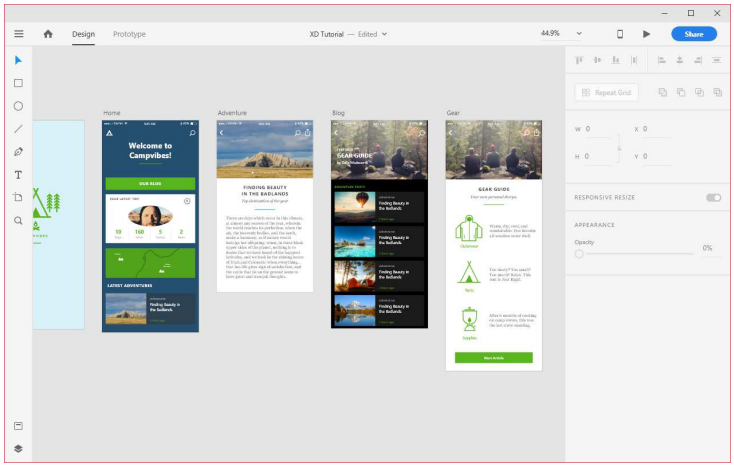


Figure 2.7: Basic Design view of Adobe XD CC native program displaying the minimalistic layout with toolbar on the left and inspection panel on the right side

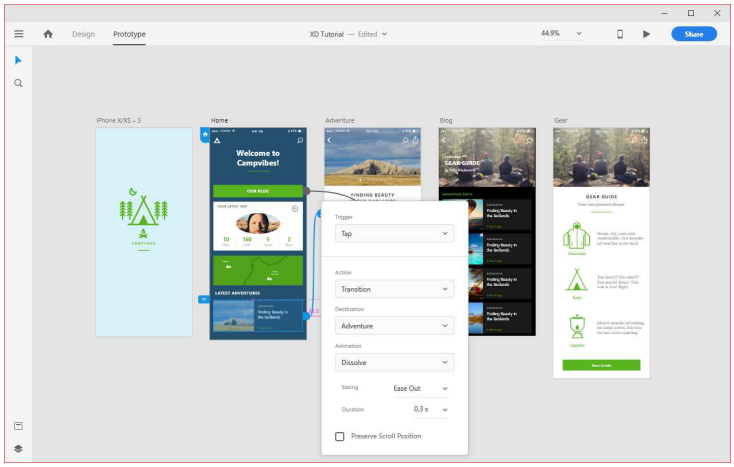


Figure 2.8: Basic Prototype view of Adobe XD CC native program displaying the different art boards and interaction dialog for creating a link between elements

Figure 2.9 shows the preview functionality of Adobe XD. Two different modes are offered, a simple interactable prototype preview for collaboration, where comments can be added to different views, and a designer spec preview for developers shown in figure Figure 2.10. Both of these preview modes generate a web link, which can be shared and also password protected. The

preview does not support mobile gestures such as swiping. Adobe XD does not offer an export option as HTML, but the developer preview does offer a lot of information for developers.

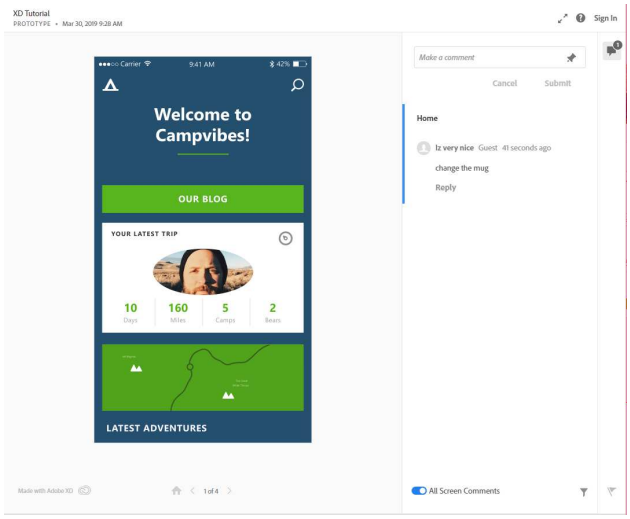


Figure 2.9: Online preview for Adobe XD CC prototype with collaboration comments on the right pane.

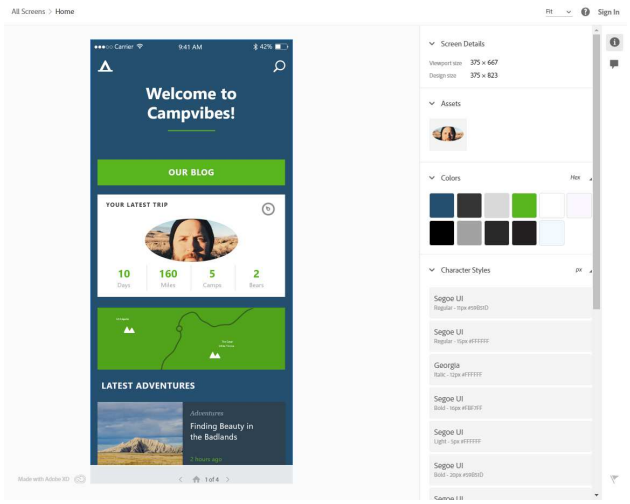


Figure 2.10: Developer preview for Adobe XD CC prototype with element information e.g. dimensions, fonts, colors and icons used on the right pane.

2.1.2.4 InVision

InVision differs the most from others, as it doesn't offer any tools to actually design a prototype. While the application is called a "digital product design platform", it can only be used to link images or parts of images to different images. InVision was selected partly because of its differences with other prototyping tools, but also since it often comes up in discussions about prototyping.

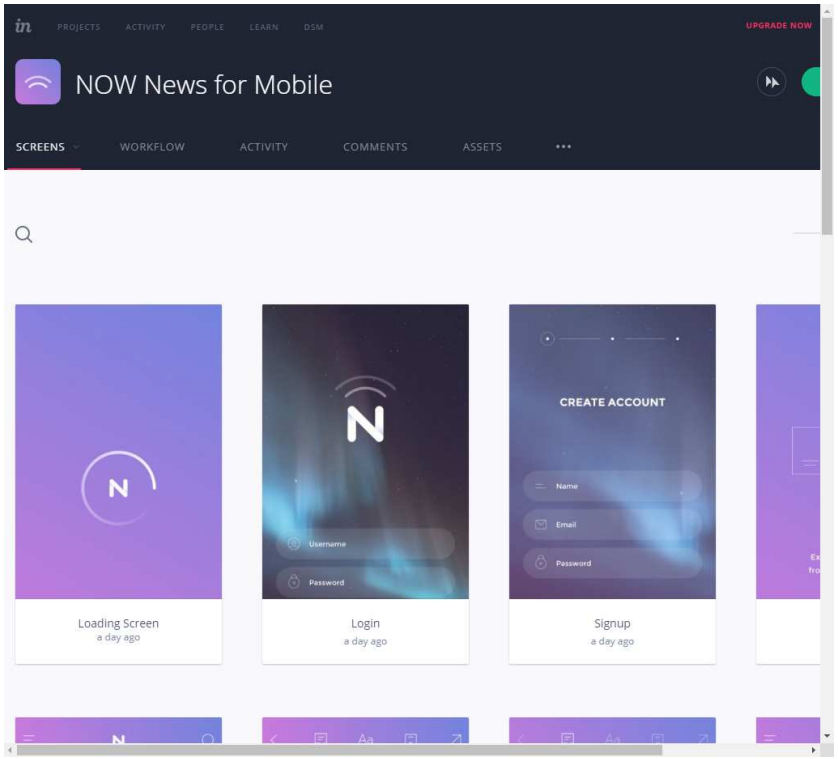


Figure 2.11: Main view of InVision web application with screens shown.

Figrue 2.11 shows the main view of the web application, displaying different created screens. These screens can not be created with the app, but must be created elsewhere and imported to InVision. Clicking the screens takes the user straight into "interactive" mode, which allows the screen to be tested. The only tool for altering the premade graphics is a simple, freehand drawing tool. No other way to change or modify the premade screens exists in the application.

Figure 2.12 shows InVisions very simplistic view on creating interactions. The user can select regions of the screen, and attach a simple link to take

the user to another screen upon clicking. There are some choosable gestures e.g. tap and swipe for triggering the event, but not very much customization is available.

Figure 2.13 shows the interactive preview mode for InVision. Comments can be added via another button, which can be tied to specific parts of the screen. A link for "sharing" to mobile can also be created in this view. InVision's preview does support the generic mobile gestures. Pressing select highlights all the available interaction zones, which is an useful feature.

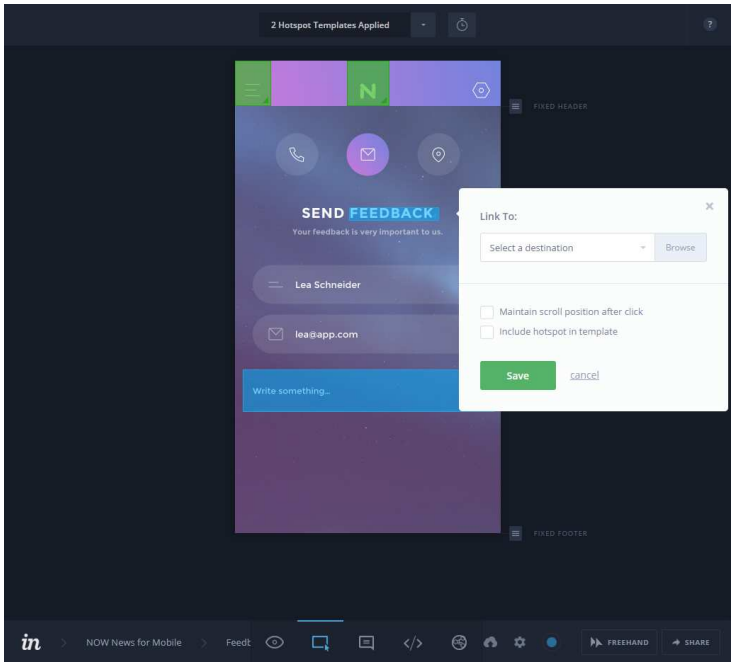


Figure 2.12: Online preview for Adobe XD CC prototype with collaboration comments on the right pane.

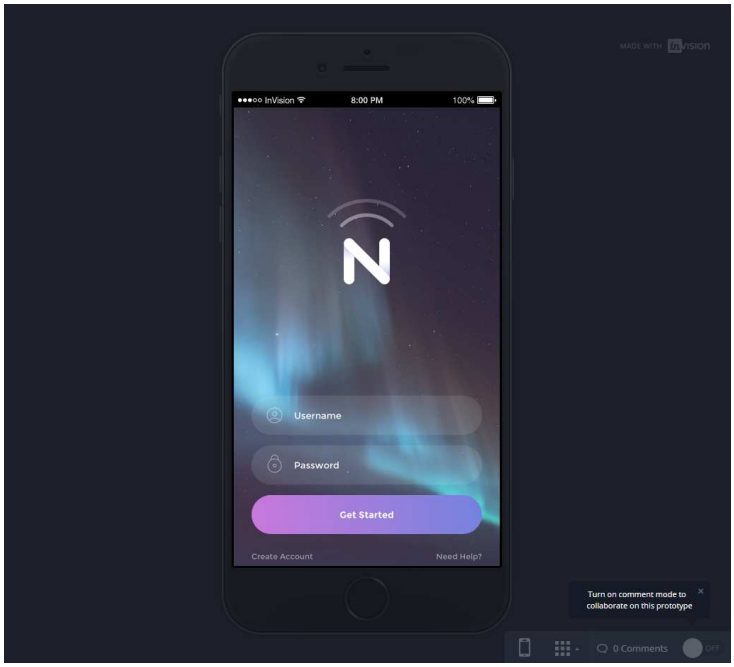


Figure 2.13: Online preview for Adobe XD CC prototype with collaboration comments on the right pane.

2.1.2.5 Prototyping tool comparison

While many of the tools varied by nature, they all offered some fresh input for the requirements and recommendations for MitMUI's development. As can be seen from the table 2.1, the tools offered much higher graphical and functional fidelities compared to traditional pen and paper prototyping. Collaboration is another section where online tools shine, as it makes development teams' cooperation much easier. Pen and paper prototyping is added to the table 2.1 as a form of control, to make comparisons between them easier. The table is not intended to be comprehensive, instead it provides an outlook to the different features available for prototyping software. The results were gathered mainly by experimenting and using the tools, as well as online review platforms aimed at developers and designers (G2crowd, 2019), (Cuello, 2019). The row titles correspond to the following explanations:

- Software: On what platform the software runs. Native here means Windows and Mac based operating systems.
- Platforms: For which platforms i.e. mobile, tablet, native etc. the tool can be used.

- Func. fidelity: Maximum attainable functional fidelity with the prototyping tool. This refers to the amount of interaction and functionality available for the tool.
- Graph. fidelity: Maxium attainable graphical fidelity with the prototyping tool. This refers to the graphical refinement of the created prototype. This is mainly affected by the available components and assets, e.g. buttons, icons etc.
- Ease of use: How easy it was to start using the application. This is metric is somewhat subjective, and susceptible for bias, although verification was attained from the online review platforms.
- Collaboration: How well the created prototypes can be shared to others.
- Generates code: Does the application generate relevant code for developers, who intent to implement the design.
- Target testing: Can the prototype be tested/shared on the destination platform, e.g. on mobile devices.

	Proto.io	Axure RP	Adobe XD	InVision	P&P
Software	Web	Native	Native	Web	-
Platforms	Mobile	All	All	All	All
Func. fidelity	High	High	Medium	Medium	Low
Graph. fidelity	Medium	Medium	High	None*	Low
Ease of use	High	Medium	Medium	High	Medium
Collaboration	Medium	High	High	Medium	Low
Generates code	Some	Yes**	Some	None	None
Target testing	Yes	Yes***	Yes	Yes***	No

Table 2.1: Four prototyping tools' features compared to pen and paper prototyping

* The graphical fidelity of InVision is listed as None, since the application does not allow building the screens inside the application. It does support as detailed screens as can be created.
** Axure RP is able to show css-like information about the prototype for developers. It does not generate HTML.
*** Axure RP and Envision both use the mobile web browser to "test" the prototype on a mobile device. Proto.io and Adobe XD can be used to natively run the prototype on a mobile device.

As can be seen from the table 2.1, all of the tested tools were relatively equal with the exception of InVision, which didn't offer any functionality for actually creating the available screens/designs. If the features would be scored from 1 to 3 with "Low" or "No" being 1 and "High" or "Yes" being 3, a best tool could be found based on these rankings. For these metrics, both Axure RP and Adobe XD would receive equal score. Their features will be taken into account when further refining the concept of MitMUI.

2.2 Web Technologies

Web technologies refers to the multitude of languages, libraries, frameworks and packages that are used in conjunction with one another for the purpose of producing web sites. Web technologies as field has seen a rise in the recent years, which reflects the ever increasing number of web applications, services and sites.

2.2.1 Web Applications

Software applications have been rapidly transitioning to the web during the last decade (Rocheleau, 2017). With the advent of HTML 5, an updated (and newest) version of the markup language, even more functionalities are supported natively inside browsers without extensive hacking and scripting. HTML 5 brought features such as the canvas, drag and dropping, web storage, and a multitude of media-centered technologies such as geolocation and web audio to the browser. Note that many of these technologies were possible to be scripted by using different methods but HTML 5 nonetheless brought a distinguished paradigm shift to the web. Many recognized software producers have also started to migrate their services to the web as shown by Rocheleau (2017).

2.2.2 HTML

HTML or Hypertext Markup Language is the standard language for developing web sites since 1993 (Connolly, 1995). It has gone through five bigger revisions as versions 1 through 5. HTML 2.0 was the first version to be standardized in 1995, and this version would serve as the baseline for all future revisions. In 2000, HTML was made the international standard as well. HTML5 is especially aimed towards the development of robust web applications (Pilgrim, 2010).

HTML describes the structure of any given website in the form of Document Object Model (DOM). DOM is created by the browser upon loading the HTML files. It consists of a hierarchy of elements identified usually as siblings or children. This DOM can be further accessed and modified by using javascript.

2.2.3 Javascript

Javascript is a dynamic, high-level, interpreted programming language. It is the essential part of web applications. While sometimes called scripting language, it still offers a very robust set of functionalities, comparable to many other programming languages. With the introduction of EcmaScript 6, classes, modules, typed arrays, collections, promises and more were made available for JS programming. Flanagan (2001) Ecma International (2018)

Javascript is traditionally used solely for client side programming, but recently has seen additions such as NodeJS which offers server-side environment to be developed with Javascript. Despite this, Javascript has traditionally been the programming language used in tandem with web development. There exists some new alternatives to Javascript for web development, but in the end, the resulting code is transpiled into Javascript. (Tilkov and Vinoski, 2010)

2.2.4 iFrame

The iFrame element allows to display cross-site content nested into a separate, embedded frame. The loaded content can be navigated separately from the main page. Since its standardization in HTML 4 and evolution in HTML 5, the iFrame element has become more robust than its predecessors. In addition to the source, name and size attributes, HTML 5 added a multitude of security-related attributes in the form of sandbox.

- allow-forms

- allow-pointer-lock
- allow-popups
- allow-same-origin
- allow-scripts
- allow-top-navigation

These include blocking nested scripts from running, preventing form submissions or APIs from loading etc. (West et al., 2016)

iFrames are predominantly used for embedding third-party media, services or ads into the document. While the iFrame could theoretically host anything (barring cross-origin policies), overpopulating a website with impractical iFrame elements might lead to bandwidth issues (Raggett et al., 1999).

iFrames are often used as tools for a multitude of web security threats (Wassermann and Su, 2008). One of the most used threats is cross site scripting (XSS), where malicious client side scripts are injected into a publicly used website, thus triggering the script for unsuspecting users (Freedman et al., 2017). iFrames are a perfect candidate to run cross site scripts, as they can be completely hidden and can access any malicious sites explicitly designed for this. While same-origin policy (SOP) mitigates this to some degree, multiple ways to circumvent this exist (Vogt et al., 2007).

Chapter 3

MitMUI - The Concept

MitMui, Man in the Middle User Interface attempts to create a new type of prototyping medium. By using an iFrame as the basis for the whole site, live production data can be used as the foundation for new features or changes.

The concept of MitMUI is very simple, create an empty HTML file as the basis for the new functionality to implement. The first element on the page is an iFrame, with its source set to the domain to use as the background. Using JavaScript, dynamically resize the iFrame to fill the whole screen, and as an added bonus, use CSS to disable the borders of the iFrame for a completely seamless experience.

The developer or designer has access to all the underlying HTML and styling via the browsers developer tools, and can start to develop the site further. The most simple use case (as was in this thesis) is the addition of an absolutely positioned element, for example a chat interface or toolbar.

While the elemental concept of the MitMUI approach is very bare boned, we can further develop the idea for a standalone framework for prototyping and developing. For this, other comparable prototyping tools will be used as reference for functionality and requirements. As no existing literature was found relating to this type of approach, this thesis is free to explore various options.

3.1 Hierarchy

At the heart of MitMUI is the iFrame. While the iFrame is a powerful tool, and especially useful for this application, it is not without flaws. iFrames are notoriously known for being susceptible for XSS attacks (Roberts-Morpeth and Ellman, 2010). Furthermore, a site's admin can a cross-origin access header to only allow the page to be accessible from inside the domain. This

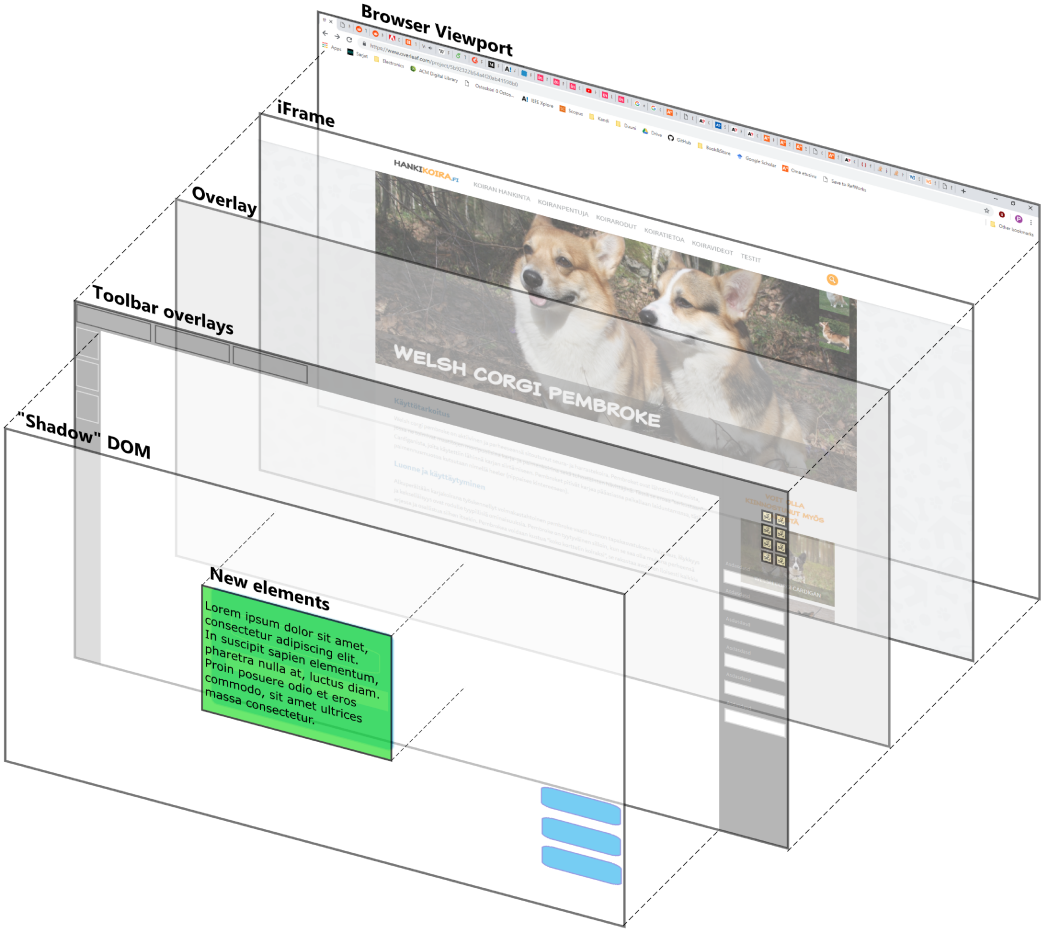


Figure 3.1: Concept image for a MitMUI prototyping tool

seriously harms the usage possibilities. But, providing that this is a prototyping tool, presumably used inside the company, it would not be too far-fetched to assume that an exception could be added to the server's configuration for the designer or developer for internal prototyping needs.

In addition to cross-origin headers, iFrames are susceptible for X-Frame-Options. These are server-side options, which permit or prohibit the content to be shown in a HTML frame, iFrame or embed element (Gondrom and Ross, 2013). HTML5 also defined the *sandbox* attribute for iFrames, which functions to mitigate the risk of including untrusted content in an iFrame. (West et al., 2016)

When applying the MitMUI approach as a prototyping tool web application, after the iFrame comes an overlay. This is to disable or enable clicks

through to the destination site, and also to toggle its visibility. On top of the overlay are different toolbars and panes for creating the actual prototype, and controlling the iFrame and overlay. Finally, the shadow DOM lies within the toolbar overlay's window (Ito, 2019). This is where the elements for a new prototype will dragged and dropped, and a new design is created. Figure 3.1 further elaborates this hierarchy.

3.2 Environment

As was mentioned in chapter 2, most of the popular prototyping tools are web-based applications. This is understandable, as it allows the usage to happen across devices and operating systems easily. Furthermore, collaborative functionalities are more readily available due to the prototype already existing online. Therefore, the proposed destination domain for MitMUI is also web application.

While some differences in user interfaces could be observed between the various prototyping tools, minimal and modern, simple interfaces seems to be preferred. This ties in with MitMUI's core functionality; to use existing, live background information as the basis for new (simple) additions to the site. Therefore, the MitMUI application would not need as heavy a component library or customization menus as some of the other prototyping tools.

From a technical point of view, the MitMUI tool would need to have basic nestable elements. Some other basic components would also be available, in a separate pane, not unlike in the many other prototyping tools. Other required panes in the user interface would be a simple interaction manager, and page/canvas tools. In the forefront, however would be a small textfield for inputting the background URL. This would always be shown (and resized properly) when developing the prototype.

3.3 MitMUI Prototyping Tool

Building on the MitMUI approach, a proper prototyping framework could be created. This would be akin to the examples provided in chapter 2.

MitMUI could be made available as a web based service for further developing web sites. The user would load up any already in use website as the basis of further development. The site would be shown in a live state via the iFrame. Then, using a simple drag-and-drop interface, the user could add elements into the places needed. These would be either absolutely positioned ("hovering" elements) or elements tied into the hierarchy already in

place on the site. Using simple JavaScript, it is possible to get e.g. the target container, into which a new custom element should be added.

The framework should support nested elements, and dynamic styling inside the web application to quickly show the end result for the user. The user could either attach event flags for elements that would need some functionality, or even code the functionality inside the web application. This way, in the end, the user would be able to get ready HTML, CSS and JavaScript files. Depending on the need, these could simply be added into the original site, or used as additional files to include into the main HTML file.

3.4 Library

While a polished prototyping tool would possibly prove especially useful for designers, an elementary JavaScript library might be more convenient for developers. After including the script into the web page in development, a simple prompt or toolbar would be visible on the page, and an iFrame as the background for the whole page. Through the toolbar, the iFrame's source could be changed and other functionalities altered.

It would be important, to on one hand to allow the iFrame to receive clicks to offer a seamless integration between the destination site and the developed feature, and on the other, disable the clicks to allow for easy testing of the new feature.

3.5 Functionality

The emphasis for MitMUI Tool is to offer developers quick ways to implement simple changes to an existing page. Therefore, the focus is to produce ready to use code. The application would take all the designed elements, and turn them into a HTML snippet. Interactions would be converted into a JavaScript template, and styling into CSS. Since the focus is to deliver the complete code for the design, it might somewhat reduce the graphical possibilities of components. Hence, the restrictions for the graphical fidelity presented earlier.

3.6 Security

While the iFrame is not inherently insecure, there exists a multitude of ways to undermine the security of it. If provided with the ability to inject custom code inside an iFrame, or the ability to change the source attribute of an

iFrame on a site, a multitude of problems and security threats are imminent. Most of these pertain to clickjacking or showing malicious fake information inside the compromised iFrame, without the user actually knowing.

Since the iFrame displays its content inside a container, the URL of the site doesn't change to reflect changes in the content. This makes it hard for users to determine if something potentially dangerous is happening inside an embedded iFrame. (Wassermann and Su, 2008)

Furthermore, this tool should not be used for sites without their explicit consent. While every websites source code and DOM is (generally) available for everyone to see, the usage of this tool might encourage plagiarization or even make clickjacking techniques more readily available. These ethical concerns will be discussed in more detail in the chapter 5.

3.7 CORS

Cross origin resource sharing ultimately defines what resources can be shared with which sites. *Origin* is an important concept when dealing with cross-origin resources. It is the defining factor when deciding whether to allow one domain's resources displayed in another domain's iFrame, for example. It is worth noting, that the origin differs with different protocols (HTTP or HTTPS) and ports used Barth (2011).

By setting the Access-Control-Allow-Origin header to '*' (all), the server informs that its content can be used under any domain (origin). This can be seen generally as a bad approach, since this once again allows for clickjacking attacks by embedding the sites content into a potentially dangerous site. Often the header is set to the specific origin, where the content originated. A great example of what constitutes a different origin is provided by Barth (2011):

```
"All of the following resources have the same origin:
 http://example.com/
 http://example.com:80/
 http://example.com/path/file
 Each of the URIs has the same scheme, host, and port
 components.
 Each of the following resources has a different origin
 from the
 others.
 http://example.com/
 http://example.com:8080/
```

```
http://www.example.com/
https://example.com:80/
https://example.com/
http://example.org/
http://ietf.org/
In each case, at least one of the scheme, host, and port
component will differ from the others in the list"
```

This means, that not all websites are inherently displayable by the MitMUI's iFrame. It depends on the destination site, and its access control settings. However, it is easy to to request that your domain be added to the list of allowed origins, if the developer of the site is e.g. a development partner or a customer etc.

3.8 Constraints / Conventions

This thesis will not present a ready framework using the MitMUI approach. Instead, the potential framework will be constructed and analyzed. User evaluations will also be presented based on the prototyped framework.

Furthermore, this approach will have its limitations. It is extremely important that this approach should mainly be used for developing web sites further internally. This means that the developer/designer should have access to the original site, or at the very least, be able to allow the development domain to access the destination domain from an iFrame due to cross origin access policy.

Chapter 4

Results and Evaluation

This chapter presents the results gathered from structured interviews. Structured interview as a qualitative method was chosen due to its reliability for acquiring desired information, and the possibility to aggregate and compare the answers inbetween interviewees (Guion et al., 2001). Furthermore, with such a limited sampling of interviewees, the structured interview was more dependable compared to a semi-structured interview. Compared to a survey, more information was hoped to be extracted for further qualitative analysis using open-ended questions and allowing free-flowing conversation (Lindlof and Taylor, 2017). While there was a significant amount of "free" conversation, all the interview questions structured the interview and kept the interviewees focused. The interview data was gathered from six interviews (first one acting as a pilot) using a recorder and interview notes. The audio from all interviews was then transcribed and analyzed by methods of content analysis.

The interview was divided into two parts, questions about prototyping tools in general, and comments and feedback for the proposed MitMUI concept. Between these two sections, a poster with images and short text was introduced to the interviewees. This poster is provided in appendix B. The interviewees had a few minutes to familiarize themselves with the concept, and had the opportunity to ask follow-up questions was something unclear. Usually the concept was elaborated again shortly, to verify that the participants had understood and could provide premeditated feedback in the second section. The interview also included a question whether the interviewee would be interested in using the product, were it developed. The 14 interview questions can be seen in appendix A.

The interviews were held in small meeting rooms. The equipment included an audio recorder, interview questions for the interviewer and the MitMUI concept poster. The interviews were conducted face to face and in

privacy in an attempt to create a more relaxed atmosphere. The mean duration for the interviews was 15 minutes, with the pilot taking only 10 minutes, and the longest interview 30 minutes. The main focus was to find more information to research questions two and three i.e. "What is a practical way of implementing a prototyping tool?" and "How does the MitMUI approach perform compared to other methods?". While some insight was gained to all research questions in the previous chapters, acquiring objective data from real developers and designers was the focal point of this thesis.

After the interviews had been transcribed, a crude spreadsheet was created with condensed and crystallized answers to all questions by the interviewees. While not a necessity for content analysis which was to follow, it served as a quick reference and familiarization for the material. This sheet is provided in appendix C. The sheet also shows that the participants consisted of 3 developers and 3 designers, aged 24-34 with varying levels of experience in their own field. The prerequisite for participation was at least some experience with any prototyping tool, and at least one year working experience either in software/web development or design.

While the semi-structured interview process already provided some elementary coding for the material, the data was further disseminated through the coding process, as much of the information was scattered along a few different answers. As the interview was divided into two sections, the data was naturally divided into two themes: pros and cons of prototyping tools and pros and cons of the MitMUI approach. The top level categories under which the codings were classified and will be presented next are:

- Pros of prototyping tools
- Cons of prototyping tools
- Positive qualities about MitMUI
- Negative qualities about MitMUI

As the pros and cons of prototyping tools could be specific to some tool, or general features of prototyping tools, the features which were answered for question "Please describe the perfect prototyping tool" were predominantly coded into the pros of prototyping tools.

In total, 161 codings were analyzed from the interview transcriptions. After the codings were coded under the four top level categories, an open card sort was performed for them. Open card sorting was chosen for it's quickness and simplicity to perform and the ability to name to groups (subcategories)

that will form (Stephanidis, 2012). While open card sorts are usually performed by a number of individuals to produce a higher realibility for the results (Nielsen, 2004), the method was performed solo for this thesis. This allowed the creation of reasonably named groups. The repercussions of this method will be discussed further in section 4.3.

Groups were created based on the content of codings. Starting from the first coding, a new group was formed with a tentative name for the group. If the next coding didn't fit into the already created group, a new one was created. This was continued until all the codings belonged to a group. In practise, this sorting was done in a spreadsheet after the method described by Spencer and Warfel (2004). The card sorting was performed three times in total, while maintaining the groups created in the first sorting with the option to create new ones. During these, some codings were discarded as too vague or just mistakenly coded and some codings were moved from very specific groups to a more general one.

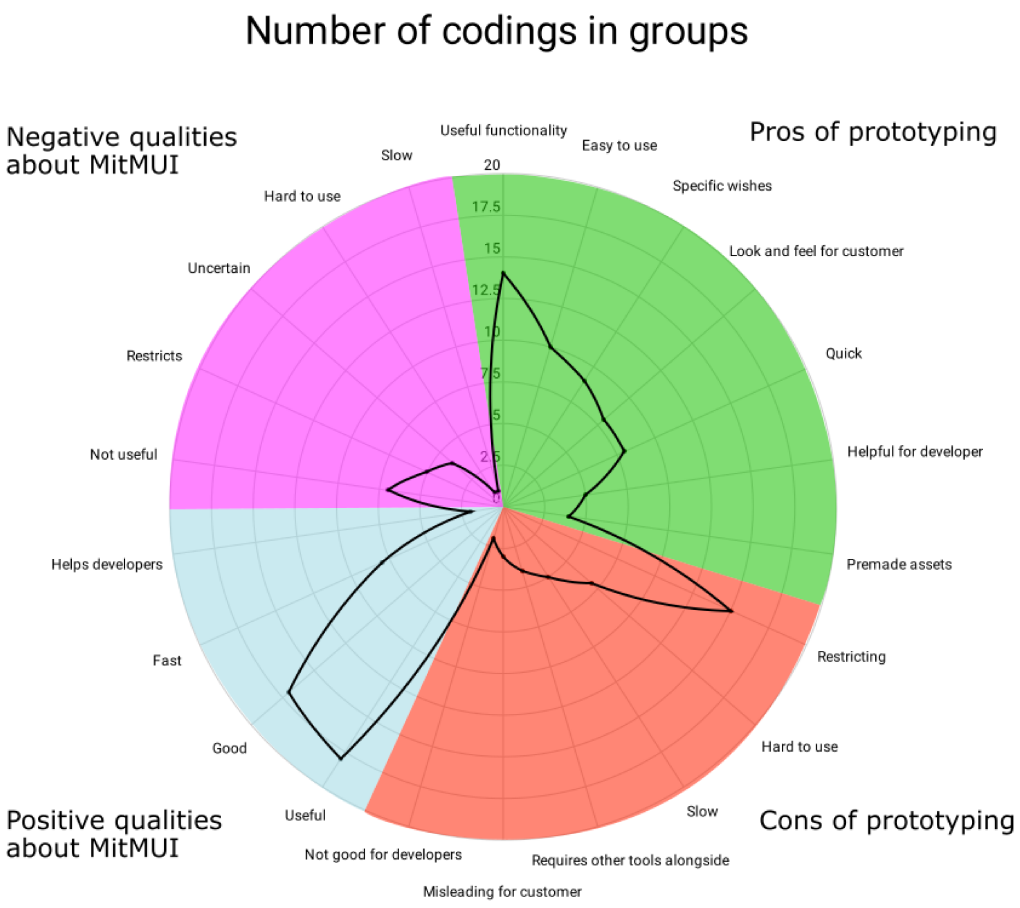


Figure 4.1: Overview of the distribution of codings under different categories and groups

With this interview and analysis, I attempted to find answers for the research questions two and three, and to further fortify some previous findings related to these. While the interview was especially formulated to get straight answers to these, more information was revealed through further coding. An overview of the gathered codings is provided in figure 4.1, but these results will be presented in more detail in section 4.2.

4.1 Research Methods

The methods chosen above in chapter 4 were chosen due to their applicability as well as ease of use. As the data to be gathered was qualitative in nature, an interview could extract the data properly and with ease (Cunliffe, 2011). The choice to use structured interviews as opposed to semi-structured or even unstructured was to guarantee that quality data could be acquired (Wood, 1997). The more rigid structured interview usually lessens the amount of material to be transcribed, and hence is faster (Wilson, 2014). As the questions that needed answering were known, the structured interview format was chosen (Cohen D, 2006). Content analysis was also sufficient to analyze the results compared to more intricate methods e.g. thematic content analysis, as the interviews were short and collected data minor.

Card sorting as a method is often used for categorizing values, hence the name (Stephanidis, 2012). As it was known that the codings needed to be separated into several groups, but the groups didn't have pre-assigned names, open card sorting provided the necessary framework for that (Spencer and Warfel, 2004). Card sorting as a method is intuitive and fast, although usually performed by a number of participants to assure small error margins (Nielsen, 2004). However, as said, the amount of data was so minute, and groups very self-explanatory, one person sufficed in this thesis. Furthermore the process was repeated three times, to guarantee that no mistakes were made, and to further reflect on the proper groupings.

4.2 Research Results

This chapter runs though the results based on the previously shown categories. Additionally, a summary and reflection about the reliability of the study will be discussed.

As stated earlier, the material was coded into four distinct categories. These all were also questions asked explicitly during the interview, but after transcription the material was found out to be very incoherent and information scattered throughout. The interview consisted of 14 questions with open discussion between them. Two of the questions however, were omitted as either redundant or non-informative. The validity of the research methods will be further discussed in section 4.3.

In appendix C crystallized answers to all the questions can be seen based on the transcriptions. It is worthwhile to notice that the first interview acted as a pilot interview for the applicability of the milieu and questions. These answers will no be taken into account in the proceeding sections, as the

interviewee's experience was lacking, but is being shown here for completion's sake. From the appendix, it can be quickly seen that all six interviewees would welcome the concept and were interested in trying it.

Also important to note is that all of the interviewees had used InVision, and out of those, two interviewees had mainly only used that. This might skew the benefits and drawbacks of prototype tools to reflect more on only InVision.

In the following sections, quotes from the transcribed interviews will be given as footnotes, so as to not clutter the page. The attributions (I1 - I6) correspond to the interviewees labeled in appendix C. These are not to mixed with bibliography sources.

4.2.1 Pros of prototyping tools

This section categorizes the positive aspects associated with prototyping tools. In addition to the crystallized answers provided in the appendix C, the good qualities of prototyping tools were categorized under seven frequently appearing groups shown in table 4.1. The groups were relatively easily named during the open card sorting, although the "Premade assets" group was included in the "Useful functionality" group during the first sorting. However, as it gained four distinct codings, it was separated into its own group.

Most of the categories are trivial to understand, but the last one might not. Since during the interviews, a plethora of very specific wishes for functionalities or features emerged, a separate group was created for those. Number of codings attributed to different categories can be seen in table 4.1. The number in parantheses refers to the number of people who had codings in that group. The following categories were (sorted by the amount of codings):

Category	Number of codings
Useful functionality	14 (4)
Easy to use	10 (4)
Specific wishes	9 (4)
Look and feel for customer	8 (5)
Quick	8 (3)
Helpful for developer	5 (3)
Premade assets	4 (4)

Table 4.1: Benefits of prototyping codings and categories

The group with most codings is useful functionality. Especially interac-

tivity and animations were brought up often [1] [2]. Other things brought up in this mixed category were the ability to have plugin support, the ability to share prototypes quickly, especially via internet link, and overall the need for "more" [3]. There was one explicitly contradicting coding, however, stating that a more simple, less feature-heavy approach would be better [4].

Ease of use also scored high, having 10 codings fall under it. This would seem to be very trivial as stated by the previous quote. With the plethora of available choices for prototyping, users want to quickly have the software up and running and creating prototypes [5]. Designers especially might veer away from using a software, if it's design is cluttered and confusing.

Other groups included the benefit of displaying a customer quickly the look and feel of a service in development, without actually having to code anything [6] [7]. This obviously provides resource benefits to all parties, customer and developers as discussed in chapter 2. Also, equally as useful as presenting to the customer was the swiftness of prototyping tools with 8 codings. The ability to quickly start designing rough sketches was given merit [8].

One of the smallest groups is the need for premade assets. Many felt that a good prototype tool should have its own library of assets i.e. different elements like buttons [9]. Here the abundance of InVision users definitely shows, as InVision does not offer any wireframing possibilities, but relies only on elsewhere created, imported graphics. The benefit of the developers also had only five codings, but they were mentioned by three individual interviewees [10].

Finally, the specific wishes group included a potpourri of various functionalities or features that were deemed useful. The main difference from the useful functionality -group, is that these were more specific, or clearly not available currently. Wishes included cross-platform support [11], more intricate animations and transitions, and larger components or regions which

[1] "... in good tools especially the fact that you can create animations ..." -I3

[2] "The possibility to create stronger transitions." -I2

[3] "I'd like the possibility to make just a little bit more." -I4

[4] "...a very simple program, no crazy amount of features so it's easy to get started." -I4

[5] "It was easy as a beginner to get started." -Interviewee 6

[6] "... you get a feel for what the user experience would be." -I3

[7] "... from the perspective of the customer, it brings about a certain wow-factor." - I5

[8] "... one can create for example really quick sketches." -I4

[9] "... that it would have some ready made components that you could drag and drop." -I2

[10] As a front-end developer, it would be great to just be able to copy all, copy and paste." - I4

[11] "... a software that would work on windows in addition to mac ..." -I6

could be available as a library [12].

Overall, it would seem that almost all of the interviewees wanted something more. When asked to describe the perfect prototyping tool, half of the interviewees stated that it would be a combination of several different tools. During the interviews a strong disdain was brought up about having to use several different tools to achieve the result.

4.2.2 Cons of prototyping tools

This section analyzes the drawbacks or lack of functionality present in prototyping tools. As before, the codings were divided into six relevant groups based on the common denominator. The groups, sorted by number of codings are shown in table 4.2. In this category, the odd one out was the "Requires other tools alongside" group. At first, this seemed somewhat disconnected from the other groups, but as before, it was included for its distinct codings and due to the fact that the thought was echoed in other interviews as well, but not explicitly enough to warrant a coding.

Category	Number of codings
Restricting	15 (3)
Hard to use	7 (4)
Slow	5 (2)
Requires other tools alongside	4 (2)
Misleading for customer	3 (1)
Not good for developers	2 (1)

Table 4.2: Drawbacks of prototyping codings and categories

The group with the most codings, "Restricting" describes the the lack of features to properly express the designer. Many felt that the used prototyping tools inhibit them from creating exactly the vision they have. Once again, the multitude of InVision users might contribute to the high number of complaints, as it does not allow to create the wireframe [13]. From the 15 codings in the category, around half were centered around the lack of assets or the possibility to create designs inside the tool [14]. Platform dependency

[12]"... you can do sideways scrolling and demonstrate all different kinds of elements." -I2

[13]"... it doesn't have the components." -I2

[14]"InVision has a freehand sketching tool, but it only allows you to draw squiggly lines." -I4

was also mentioned as a drawback for some native tools [15].

Second highest group is "Hard to use". This included problems from the lack of useful features [16] to being outright too hard to get started [17]. This category also had the highest number of individual mentions, so it would seem to be a very prominent drawback in prototyping tools and that user experience plays a key role when deciding on a tool [18].

Group "Requires other tools alongside" was separated from others, as its opposite was also mentioned many time in the "Benefits of prototyping tools" category. This again was brought up in many different contexts, and not only in the drawbacks. While only two interviewees mentioned this explicitly [19], the notion was echoed through other topics [20].

Some of the smaller groups include "Slow", "Misleading for customer" and "Not good for developers". Again, though small in size, these have their logical counterpoints listed for the benefits and therefore warranted their individual categories. Since the sampling was relatively small, it's hard to say whether "Misleading for customer" and "Not good for developers" are truly as miniscule as the table 4.2 shows. It might be deducted, that designers would focus more on the client side, as they're engaged with the customers more so than developers. On the other hand the "Not good for developers" was only mentioned by one developer, as a developer might have better grounds for such a statement. A coding pertaining to the "Misleading for customer" -group describes the difficulty in showing a non-tech-savvy client a prototype of a service [21]. A more developer-friendly approach to prototyping tools is wished, as many of the current tools do not offer any kind of usable code for developers [22].

The sentiment about slow prototyping tools mostly showed the confusion about why some things might be very fast to prototype, while others are very tedious to implement [23]. Other coding categorized as "Slow" pertained to the physical limitations of web-based prototyping tools, as "it started to

[15]"... it limits the selection pretty hard, since many of those programs are in principle for mac." -I6

[16]"I had to count pixels and then guess whether it would go a bit to left or a bit to the right." -I5

[17]"... a little hard to get started." -I6

[18]"... it had some technical difficulties." -I2

[19]"... it shouldn't be like you design somewhere, prototype elsewhere and then develop in yet another program." -I2

[20]"None of the tools have been kinda all-in-one solution" -I4

[21]"... people who don't realize what a prototype is assume that it's already a functional website. Like 'What do you mean you need to create this again'" -I2

[22]"A more developer-friendly approach would be nice." -I4

[23]"... might take a surprisingly long time ..." -I2

have so many elements, that it slowed down." [Interviewee 6].

The overall sentiment seemed to be, that many prototyping tools are too limiting or hard to use. Once again, the wish for an all-in-one solution was present. The concerns in this category, however, would seem to be mitigated with MitMUI, as ease of use is one of its focal points along with the lessened need for multiple tools during a services lifespan from designing to development.

4.2.3 Positive qualities about MitMUI

This section presents the good qualities about the MitMUI approach that were discussed. The content analysis of this section was somewhat more difficult, as some interviewees found it hard to analyze some things on a conceptual level. This issue will be further discussed in section 4.4. While "Good" is fairly uninformative category, it was the only one that fit many of the codings. The other three groups are again reflected in the section 4.2.1 groups. The categories ordered by the amount of codings are shown in table 4.3.

Category	Number of codings
Useful	18 (4)
Good	17 (5)
Fast	8 (3)
Helps developers	2 (1)

Table 4.3: Positive qualities of MitMUI approach codings and categories

The highest scoring group was "Useful". While some of the codings were a bit difficult to divide between the "Good" and "Useful" categories, some also explicitly touted the usefulness of this approach [24]. Few codings went into further detail about why this approach would be useful for them. The ability to get ready made code was praised [25], as well as the concept's utility for further development [26].

The conceptualized approach received very good praises in the form of codings under the "Good" category as well. About half of the codings in this category are plain "sounds good" responses, but some go to more detail about the benefits. This category will also be used when determining whether

[24] "You get a working prototype at the same time so that's wildly useful." -I5

[25] "If you can paste the code into somewhere easily then that's a very good feature." -I6

[26] "It would be very useful especially for further development." -I4

the MitMUI approach is a viable one for prototyping tools. Noteworthy is interviewee 2's reaction [27], with the intent of echoing their previous comments during the "describe a perfect prototyping tool" -question. In other words the MitMUI prototyping tool described to them perfectly corresponded to their needs and wishes. Other codings included more general praise from all the participants [28] [29] [30] [31].

A more elaborate coding pertains to how during further development e.g. in the case of a new customer with existing site, it is usually time consuming and tedious to create a copy of the existing site only for prototyping purposes [32].

The proposed approach also received a good amount of codings for the "Fast" group [33]. Many of these sentiments again referred to the further development done for an existing site, but benefits for the customers were also mentioned [34]. Also the fact that with this approach, the prototyping wouldn't have to be started from scratch received praise [35].

Finally, coming in last place in terms of codings is "Helps developers". Although a bit disappointing, as one of the main foci of MitMUI was that it would provide ready made code for developers more advantageously, this benefit might not have been so clear to the designers who composed half of the interviewees. Once again, resource benefits for the customer were brought up as well [36].

With the previous quote, it's worth mentioning that throughout the whole interview, interviewee 6 used the adverb "maybe" numerous times. This uncertainty might just be ingrained with the interviewee's way of speaking, or in some cases it might have been used as a suggestion more than an adverb indicating doubt, which is often the case in spoken English Cambridge Dictionary (2019).

Overall, the proposed approach received high praises. While it was a bit disappointing, that only four categories could be created from the codings i.e. it can't be directly compared to the "Pros of prototyping tools", the

[27] "... so this would be exactly that type of thing then." -I2

[28] "I like this concept." - I4

[29] "Good idea." - I4

[30] "Sounds good." - I3

[31] "I can see it working." - I6

[32] "It would solve the problem of making small changes to websites." -I 5

[33] "Might be faster to progress with the prototype." -I6

[34] "Practically that means that it's faster done and that the customer can get it earlier." -I5

[35] "... the fact that you don't have to start from scratch." -I5

[36] ... helps the developer to quickly upload a prototype for a customer ... so maybe it will be helpful." -I6

overall feedback was very positive.

4.2.4 Negative qualitites about MitMUI

The codings of the negative qualities of MitMUI were divided into five groups, although two of them consisted of only one coding. As before, these were still considered as groups as they're reminiscent of the groups in section 4.2.2. This category has the least amount of codings altogether and the groups ordered once again by the amount of codings are shown in table 4.2.4.

Category	Number of codings
Not useful	7 (3)
Restricts	5 (3)
Uncertain	4 (1)
Hard to use	1 (1)
Slow	1 (1)

Table 4.4: Negative qualities of MitMUI approach codings and categories

Majority of the codings were codified into the "Not useful" group. While a bit perplexing, as the there was also 18 codings in the "Useful" group in the previous section, most of them addressed the issue of MitMUI being mostly useful for existing sites [37]. This is true in that the main focus of MitMUI is the ability to use already existing web site as the basis for further development, but even without a premade site, the proposed web-based prototyping tool should prove useful in general prototyping. One of the harsher critiques found it hard to see the benefits of using this [38].

One interviewee also brought up whether the MitMUI-compiled HTML, css and js code would actually prove useful. This was in reference to another pseudo-prototyping tool Zeplin, which allows the creation of static wireframes, and displays the element's properties as css [39]. Two interviewees critizised the quality of this css, as its oftentimes not all the properties an element would need to match the prototype in Zeplin.

Second highest group was "Restricts". Three interviewees voiced their concerns, whether MitMUI would end up as any other prototyping tool,

[37]"If the purpose is to design a completely new service which doesn't exist, then this is not for that." -I4

[38]"What would be a bit questionable is that how useful it would be, or rather when it would actually provide benefits" -I6

[39]I'm questioning how purposeful it is to create some Zeppelin (sic.) level CSS." -I6

where it might restrict the designers' vision [40]. Another coding in this category again displayed doubt about whether this approach would be usable outside further development [41].

One coding in the "Restricts" group also took into account the proposed web-basedness of the MitMUI prototyping tool, and the technical limitations it might cause i.e. mainly being slow compared to native software [42]. This is a valuable piece of information, as the technical limitations of web applications relating to speed were not considered before.

One interviewee is solely responsible for all the codings in the "Uncertain" group. These mostly referred to the linking created between the background site and the prototype created [43]. They voiced concerns whether this would break the prototypes as the background page would be updated, or cause more work in the form of several supported versions of the website just to keep the prototype up to speed [44]. Again, these are very valid critiques about the MitMUI approach, and would need to be mitigated somehow if implemented. These critiques will be addressed later in the section 5.

Finally, both categories "Hard to use" and "Slow" only had one coding. The hard to use concern was voiced in the context of not having any site to build on top of [45]. Other codings of this type were categorized as "Not useful" above, but as this coding included the sentiment of the MitMUI being hard to use, a separate category was created.

The slowness was critiqued by the same interviewee who had said that this approach would be faster earlier [46]. It is somewhat perplexing, and raises the concern whether the interviewee actually knew enough what they were talking about. The fault might also be in the interview itself and concept introduction, but these will be further discussed in section 4.4.

It is worth mentioning, that interviewee 2 found no negatives about this approach. Furthermore, the categories created for the negatives had less individual mentions than the positive categories. Also, the "Useful" category surpasses the "Not useful" category in codings by 11, so one might say that the overall evaluation of this approach was that it's useful.

[40]"... maybe it's better to leave some space for the wild ideas." -I5

[41]"... it limits what kinds of things can be done with this." -I4

[42]"I'm just scared that if that's being run in a browser, it might severely limit the complexity of the prototypes. Or not necessarily, surely there's a lot of optimization available for browsers, but that's just a thing that worries me - and might worry many other people as well." -I6

[43]"... how easily that would break." - I6

[44]"It's challenging to do when the page changes ..." -I6

[45]"If there's no background, then it will get pretty hard to use." - I5

[46]"Then again, I'm not sure if this would take more time." - I3

4.3 MitMUI Viability / feasibility

The results from the interview showed mostly positive outlook for the MitMUI approach. Out of the five interviewees, four people contributed 18 "Useful" codings. This makes the "Useful" group the the largest. Second highest coding group of all is "Good". Many people felt that this approach might lessen the workload and software clutter needed especially for further development of services. One interviewee even responded after the introduction of the concept, that this was everything he had been hoping for. While the proposed MitMUI approach or tools was not meant to be an "all-in-one" solution, many still expressed the value that this would bring. This was mainly due to the fact, that the prototype would be created simultaneously with designing, after which the prototype could be passed on the developer who would be able to use concrete code produced by MitMUI.

Other good qualities which were named for prototyping tools also coincide with the coded good qualities of MitMUI. Ease of use was always the goal for MitMUI, and also comes as somewhat a requirement, due to the fact that valid code must be generated for the develoeprs. As discussed before, this might pose some restrictions on the graphical fidelity available for the tool, so as to be able to decipher the generated wireframe of the prototype. While not present as a group in the "Positive qualities about MitMUI" category, the "Look and feel for customer" as well as "Helpful for developer" groups were also featured in the interviews.

The main issue found during the interviews was that this new approach might prove just as restricting as some other tools. The "Restricting" group was featured in both, "Cons of prototyping tools" and "Negative qualities about MitMUI", although in smaller size in the latter. This might have been partly due to the fact, that some interviewees didn't see value in MitMUI without an already existing site. As from the concept side, the MitMUI tool should be as usable for further development of existing sites as well as for brand new sites in that it would feature all the generic features for creating prototypes.

Other valuable feedback consisted of the limitations of web-based applications, and the quality of the provided code. During this thesis, and its limited experiments with different prototyping tools, no performance issues were found in web-based applications compared to native software. This is not to say that there might be a hard limit how elaborate designs the MitMUI could support. Further research would be needed to discover how far the browser's performance can be optimized. Also valuable was the notion, that many new modern web frameworks, e.g. React use very different approach

for creating and manipulating the DOM from traditional "pure" HTML or even HTML templating languages. This in turn may make it difficult for the MitMUI (web application) to provide usable HTML for all developers. Some information about the hierarchy would nonetheless be presented for the developers, even without support for multiple styles of produced code.

As for the information security aspect of the iFrame, no huge issues were observed. As was stated in chapter 2, a website can prevent itself being shown through an iFrame. This naturally makes the approach more safe, albeit more limiting. But since this tool was always meant to be used for quick additions to existing sites and not stealing from other websites, it does not pose such a great limitation. More ideas for future work will be provided in section 5.2.

4.4 Research reliability

While there's evidence that 5 interviewees would produce about 80% of noticeable results (Nielsen, 2000), these numbers applied explicitly to usability testing. The research could have benefited from having for example ten interviewees. Something which posed a more severe impediment for the study was that especially most of the developers didn't have expert experience on the subject. Almost all of the beneficial critique was provided by only one of the interviewees. Especially many designers hoped for a prototype to be able to provide more meaningful feedback.

The interview skeleton could have also been designed better, and more care taken that all the interviewees provided enough data for the questions. As stated earlier, even with the pilot interview, two questions turned out to be either redundant or too confusingly worded for some interviewees. However, the majority of the questions were comprehended correctly.

The diversity of the interviewees could have also been better. Out of the total of 6 interviewees (one pilot), 5 of them worked in the same place. This probably explains the prevalence of InVision in the answers. Furthermore, as the interviews progressed, two questions were deemed redundant or too confusing for the interviewees e.g. the other question about the dual tools. This shows as the few blank cells in the appendix C.

Also nonideal is that the codings were created and card sorted by only one individual. This lowers the reliability of the study somewhat, but the discovered information is still relevant. Also, as the coded groups were very mutually exclusive i.e. pros and cons, positive and negatives, the margin of error might be lower than for other more diverse or robust material.

Chapter 5

Discussion and Conclusions

The aim of this study was to research whether the MitMUI method could be used as a novel new prototyping approach. Based on the existing literature, no previous implementation of this approach were found. Therefore it seemed valuable to study why this has been the case. In relevant literature, no restrictions for the viability of this approach were found, except for the issues related to cross-origin policy.

From comparing other similar or successful prototyping tools, valuable insights were gained about the properties of prototyping tools. Many of them valued simplicity, though most even to the point of sacrificing functionality. Even though InVision seems to be one of the more popular choices for creating interactive prototypes, the author fails to see it's success due to lacking features and the need for other software.

Surprisingly few prototyping tools offered "ready to use" code. A definitive answer as to why this is the case was not found during this research. Reflecting on the interviews, the reasons might be due to the difficulties in producing said code, since some design choices might not be explicitly translatable to code, or due to the fact that the quality of producible code would be too poor. These are the main foci for further research, but these will be discussed further in section 5.2.

The research results show great promise - need even - for a new type of prototyping tools which would enable the designers to only work with one software for all designing and prototyping needs. While the interview demographic might have some bias as the majority of interviewees are engaged in further development of existing sites, the benefits of MitMUI for further development were discernible. Almost all interviewees praised MitMUI's utility when dealing with existing sites. Unfortunately some of the interviewees failed to grasp MitMUI's notion also as a general prototyping tool.

Out of the six interviewees (pilot included), all were eager to try a proto-

typing tool based on the MitMUI approach, and also said that we're they in a situation where a product would have to prototyped and developed, then this tool would seem ideal. Furthermore, the positive feedback for MitMUI was greater in number than the negative. All the polarized groups e.g. useful vs not useful, fast vs slow had more codings in the positive counterparts than negative ones. This would suggest again, that this approach shows serious promise when implemented correctly.

5.1 Answers to Research Questions

- *RQ1:* What kinds of tools are already available for prototyping web applications?

  Do any of them allow live data to be used?

  Are there any tools that would utilize iFrame?

During the research for chapter 2, it was discovered that a plethora of different tools, some better suited for specific type of prototyping exist. Four prototyping tools were chosen to be studied further, based on popularity or other merits. While not an exhaustive dissection of all the existing prototyping tools, valuable information was gathered about their features, functionality and user experience. This aided in formulating the MitMUI concept.

No existing prototyping tools which would use an iFrame or allow live background data to be used were found during the research. The usage of an iframe for prototyping also seems to be an unresearched subject. It might be, that due to the absurdness of using an iFrame, which is usually somewhat frowned upon due to it's security and performance issues, no one has even thought about utilizing one.

- *RQ2:* What makes a prototyping tool practical?

  Is the use of an iFrame applicable for a prototyping tool?

  What kind of restrictions does using an iFrame pose?

From the research performed on the previously mentioned prototyping tools, initial information about features and functionalities required was found. This information was further refined through the conducted interviews. A sizeable amount of desirable features for prototyping tools were presented in previous chapters. Out of these, ease of use, robust functionality and swiftness stood up.

Many interviewees expressed, that they would just want to be able to quickly get their designs visible without too much fine-tuning. Therefore the

easier to use a tool is, the quicker it is to use. One also has to remember the reason why prototypes are oftentimes created, which is for the potential customer. For this reason the sharing features and ease of use, especially for the customer when seeing the finished prototype, are extremely important aspects for prototyping tools.

The iFrame, while limited, would seem applicable for a prototyping tool. Based on the previous research, no fatal problems were discovered. The only limiting factors of an iFrame are it's security issues. Since iFrames are inherently so insecure, multiple different safety features have been implemented with HTML5 e.g. sandbox attributes and single-origin policy. These security features are meant, however, only for unsolicited use of an iFrame. Therefore it's restrictions can be nullified provided that the site to be shown in an iFrame will have it's content origin policies set correctly for the use of MitMUI.

- *RQ3:* How does the MitMUI approach perform compared to other methods?

Based on the interviews, the MitMUI approach seems to be a very viable solution as a prototyping tool. All of the interviewees were interested in the approach, and many concluded that it would be extremely useful. While no true comparisons could be made before a minimum viable product would be produced, the initial results show great promise for MitMUI. Based on the gathered codings and their analysis, the benefits of the MitMUI approach "outperformed" all the other categories i.e. one could argue that it is even better than existing prototyping tools. It was also easy to see the it received significantly more praise than criticism in the form of codings, even if some of the results might have been biased.

5.2 Future Work

The next obvious step is to create a minimum viable product (MVP) of MitMUI, and conduct usability tests for the prototype. The approach seems to warrant a product, so a prototype should be made. Based on the valuable information provided by the interviews, care should be taken when implementing the MitMUI prototyping tool as a web application. The limitations of web applications might pose some problems. More research is still needed for discerning whether the browser-based solution can be optimized in such a way, that performance wouldn't be an issue.

Furthermore, if this new approach would be made into a commercial product, it might prove beneficial to improve MitMUI's general prototyping

features. Many interviewees were concerned if there would be any use for MitMUI outside of further development of websites. The ability to use an existing site as the basis is a valuable feature, but as stated by many interviewees, they would like to have an "all-in-one" solution. Therefore more research is needed to hone the MitMUI even further, and to allow it to be not just another prototyping tool, but the next big breakthrough for the prototyping world.

References

Adobe (2019). Download free adobe xd — ux/ui design and collaboration tool. Referenced 20.5.2019. Available: https://www.adobe.com/products/xd.html.

Ali, K. (2017). A study of software development life cycle process models. *International Journal of Advanced Research in Computer Science*, 8(1).

Axure (2019). Axure rp 9 - prototypes, specifications, and diagrams in one tool. Referenced 20.5.2019. Available: https://www.axure.com/.

Barth, A. (2011). The web origin concept. *No. RFC 6454. 2011.*

Berlin, T. U. (2019). Technische universität berlin. Referenced 20.5.2019. Available: https://www.tu-berlin.de/menue/home/parameter/en/.

Budde, R. and Zullighoven, H. (1990). Prototyping revisited. In *COMPEURO'90: Proceedings of the 1990 IEEE International Conference on Computer Systems and Software Engineering-Systems Engineering Aspects of Complex Computerized Systems*, pages 418–427. IEEE.

Cambridge Dictionary (2019). Maybe or may be? - english grammar today. Referenced 20.5.2019. Available: https://dictionary.cambridge.org/grammar/british-grammar/maybe-or-may-be.

Cohen D, C. B. (2006). Qualitative research guidelines project. Referenced 23.5.2019. Available: http://www.qualres.org/HomeStru-3628.html.

Connolly, D. W. (1995). Hypertext markup language-2.0. *IETF*.

Crinnion, J. (1991). *Evolutionary systems development: a practical guide to the use of prototyping within a structured systems methodology*. Pitman London. Improved and increased user involvement: Prototyping requires user involvement and allows them to see and interact with a prototype

allowing them to provide better and more complete feedback and specifications. page 18 wiki 7.

Cuello, J. (2019). Prototyping tools. Referenced 20.5.2019. Available: http://www.prototypingtools.co/.

Cunliffe, A. L. (2011). Crafting qualitative research: Morgan and smircich 30 years on. *Organizational Research Methods*, 14(4):647–673.

Davis, A. M. (1992). Operational prototyping: A new development approach. *IEEE Software*, 9(5):70–78.

DFKI (2019). Forschungszentrum für künstliche intelligenz. Referenced 20.5.2019. Available: https://www.dfki.de/en/web/.

Ecma International (2018). Ecmascript® 2018 language specification. Technical report, European Computer Manufacturers Association.

Elverum, C. W. and Welo, T. (2015). On the use of directional and incremental prototyping in the development of high novelty products: Two case studies in the automotive industry. *Journal of engineering and technology management*, 38:71–88.

Flanagan, D. (2001). *JavaScript: The Definitive Guide, 4th Edition*. O'Reilly, 4 edition.

Freedman, R., Lewis, D. A., Michels, R., Pine, D. S., Rush, A. J., Schultz, S. K., Tamminga, C. A., and Katz, R. (2017). 2017 in review. *American Journal of Psychiatry*, 174(12):1140–1143.

G2crowd (2019). Proto.io reviews 2019 — g2. Referenced 20.5.2019. Available: https://www.g2crowd.com/products/proto-io/reviews.

Gondrom, T. and Ross, D. (2013). Http header field x-frame-options. *RFC Editor - RFC 7034*. Referenced 20.5.2019. Available: https://www.rfc-editor.org/info/rfc7034.

Google Trends (2019). Interest over time - chatbot. Referenced 20.5.2019. Available: https://trends.google.fi/trends/explore?q=chatbot.

Gordon, V. S. and Bieman, J. M. (1995). Rapid prototyping: lessons learned. *IEEE software*, 12(1):85–95.

Guion, L. A., Diehl, D. C., and McDonald, D. (2001). Conducting an in-depth interview. *EDIS 2001*.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1):75–105.

Hill, J., Ford, W. R., and Farreras, I. G. (2015). Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations. *Computers in Human Behavior*, 49:245–250.

Hussain, S., Sanders, E. B.-N., and Steinert, M. (2012). Participatory design with marginalized people in developing countries: Challenges and opportunities experienced in a field study in cambodia. *International Journal of Design*, 6(2).

Ito, H. (2019). Shadow dom. *Shadow DOM, W3C Editor's Draft, April 2019*.

Lindlof, T. R. and Taylor, B. C. (2017). *Qualitative communication research methods*. Sage publications.

Martin, R. C. (2002). *Agile software development: principles, patterns, and practices*. Prentice Hall.

McCurdy, M., Connors, C., Pyrzak, G., Kanefsky, B., and Vera, A. (2006). Breaking the fidelity barrier: an examination of our current characterization of prototypes and an example of a mixed-fidelity success. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1233–1242. ACM.

Nielsen, J. (2000). Why you only need to test with 5 users. Referenced 20.5.2019. Available: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/.

Nielsen, J. (2004). Card sorting: How many users to test. *Jakob Nielsen's Alertbox*. Referenced 20.5.2019. Available: https://www.nngroup.com/articles/card-sorting-how-many-users-to-test/.

Overmyer, S. P. (1991). Revolutionary vs. evolutionary rapid prototyping: balancing software productivity and hci design concerns. *Center of Excellence in Command, Control, Communications and Intelligence (C3I), George Mason University*, 4400. wiki 8.

Pilgrim, M. (2010). *HTML5: up and running: dive into the future of web development*. " O'Reilly Media, Inc.".

Proto.io (2019). Proto.io - prototypes that feel real. Referenced 20.5.2019. Available: https://proto.io/.

Raggett, D., Le Hors, A., Jacobs, I., et al. (1999). Html 4.01 specification. *W3C recommendation*, 24.

Roberts-Morpeth, P. and Ellman, J. (2010). Some security issues for web based frameworks. In *2010 7th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP 2010)*, pages 726–731. IEEE.

Rocheleau, J. (2017). Why web apps are becoming the new native apps. Referenced 20.5.2019. Available: https://envato.com/blog/web-apps-becoming-new-native-apps/.

Rome, N. Y. (1992). Software prototyping and requirements engineering. *ITT Systems & Sciences Corporation*. Disadvantages of prototyping.

Ronacher, A. (2019). Welcome — flask (a python microframework). Referenced 20.5.2019. Available: http://flask.pocoo.org/.

Snyder, C. (2003). *Paper prototyping: The fast and easy way to design and refine user interfaces*. Morgan Kaufmann.

Spencer, D. and Warfel, T. (2004). Card sorting: a definitive guide. *Boxes and arrows*, page 2.

Stephanidis, C. (2012). The encyclopedia of human-computer interaction. *The encyclopedia of human-computer interaction*.

T-Systems Multimedia Solutions (2019). T-systems multimedia solutions: Corporate site. Referenced 20.5.2019. Available: https://www.t-systems-mms.com/en.html.

Tilkov, S. and Vinoski, S. (2010). Node. js: Using javascript to build high-performance network programs. *IEEE Internet Computing*, 14(6):80–83.

Ulrich, K. T. (2003). *Product design and development*. Tata McGraw-Hill Education.

Vogt, P., Nentwich, F., Jovanovic, N., Kirda, E., Kruegel, C., and Vigna, G. (2007). Cross site scripting prevention with dynamic data tainting and static analysis. In *NDSS*, volume 2007, page 12.

Walker, M., Takayama, L., and Landay, J. A. (2002). High-fidelity or low-fidelity, paper or computer? choosing attributes when testing web prototypes. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 46, pages 661–665. SAGE Publications Sage CA: Los Angeles, CA.

Wassermann, G. and Su, Z. (2008). Static detection of cross-site scripting vulnerabilities. In *Proceedings of the 30th international conference on Software engineering*, pages 171–180. ACM.

West, M., Barth, A., and Veditz, D. (2016). Content security policy level 2. W3C recommendation, W3C. https://www.w3.org/TR/2016/REC-CSP2-20161215/.

Wilson, C. (2014). *Semi-Structured Interviews*. Interview techniques for UX practitioners.

Wood, L. E. (1997). Semi-structured interviewing for user-centered design. *interactions*, 4(2):48–61.

Appendix A

Interview text

Hello, thank you for participating in this study about a new prototyping approach. The aim of this study is to gather data and feedback about the new proposed approach. The interview will be recorded, and answers anonymized.

Do you have front-end development experience?
If yes, how many years? Note: development experience here includes actual development work, as well as any fruitful hobby projects.

Do you have designing experience? If yes, how many years? Note: designing experience here refers to actual work as a designer and fruitful prototypes built.

Have you prototyped interfaces before? What kind?

What tools have you used for prototyping, why?

What would you say are general good qualities of prototyping tools, and any benefits to those you have specifically used?

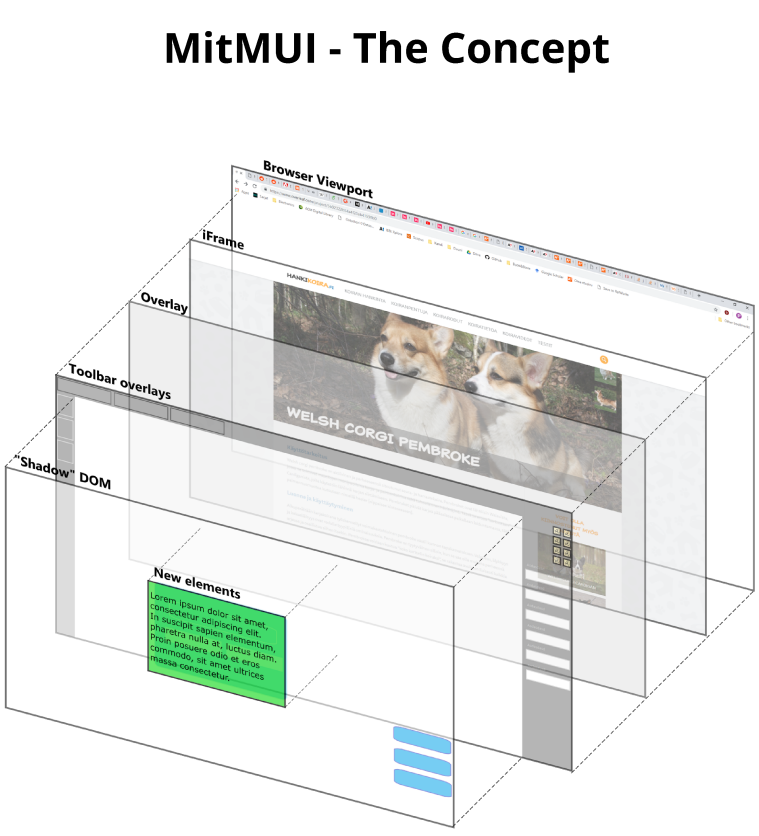
What about drawbacks?

Please describe the perfect prototyping tool. What features or functionalities it would have, what kind of a user interface and workflow?

```
.....
Here you see the concept and some illustrations
about the the new proposed prototyping approach.
.....

Do you have any questions or clarifications
about it?

What is your first impression about the new
approach?

In my thesis, I will present two different ways
of using this approach; as a standalone tool
akin to others (e.g. some you have used before?),
and a javascript library designer for developers.
In which situations would you think these would
bring extra value, and why?

What would you say are the benefits of this new
approach?

What about drawbacks?

What would you change about the approach, or the
tools?

Do you think you would use either one of these
methods as a designer/developer?

As a designer/developer, do you think that
either one of these tools would only be used
by designers/developers? Why?

Thank you for you time!
```

Appendix B

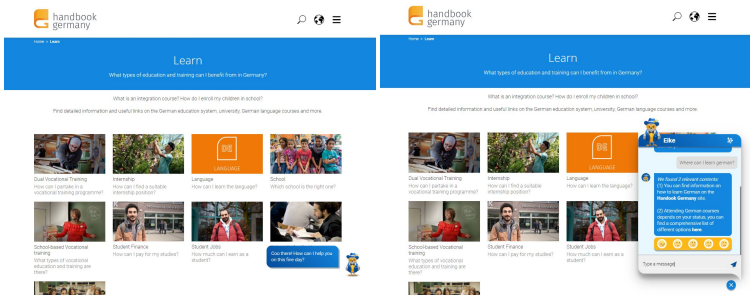
Interview MitMUI concept poster



The concept of Man in the Middle User Interface is to use an iFrame as the basis for future (web) prototyping. This allows for quick *extreme prototyping* of new features for an existing site, e.g. the chatbot shown below. The designer/developer also need not know about the underlying technology or site structure, even when using it as the basis.

The concept is meant to be applied as two separate approaches:

- As a web application akin to many others prototyping tools (Proto.io, Axure RP etc) which would allow building interfaces using drag-and-drop interface on top of (virtually) any existing website. The tool would compile the "shadow DOM" created by new elements into ready-to-use html, js and css.
- As a (javascript) library which, when included into a page, would create an iFrame as the bottom element, and add a simple interface for controlling it's behaviour. This library would be aimed predominantly for developers.



The applicability of this approach was tested in Aalto University's joint project with Handbook Germany, while developing a chatbot for their site without access to the source code. The image above shows the Handbook Germany site loaded inside an iFrame, and the actual site only has the chatbot elements, style and functionality.

Figure B.1: MitMUI concept poster for interviews

Appendix C

Interview condensed answers

Question / Interviewee	Interviewee 1, female (developer, Pilot)	Interviewee 2, male (designer)
Development experience	1 year	2 years
Design experience	None	15 years
Interface prototyping experience	Very little, school projects	Yes, websites, mobile applications
Used prototyping tools and why	None	InVision (good plugins), Adobe XD
Good qualities of prototyping tools	Don't know	Quick and easy to provide look and feel
Drawbacks of prototyping tools	Don't know	Limiting, some don't have good components, others lack functionality
Describe perfect prototyping tool	Easy to use, many useful features	A combination of InVision and some other tool. More functionality and ready drag-and-drop components for more complex interactions
First impression of MitMui	Seems better than others	Seems really good, kinda what I described earlier.
Differences / opinions about MitMui tool and library	Don't know	It's good that there's the option to choose
Benefits of new approach		The library option seems great since no other's seem to offer that
Drawbacks of new approach		Might be hard to switch the program you're used to
Would you use	Yes	Yes, interested

Question / Interviewee	Interviewee 3, female (designer)	Interviewee 4, female (developer)
Development experience	None	3 years
Design experience	2 years	1 year
Interface prototyping experience	Yes, websites, mobile applications and games	Yes, websites and mobile applications
Used prototyping tools and why	InVision, Zeplin, Unity (for games)	Photoshop, InVision, Adobe XD, Marvel.
Good qualities of prototyping tools	Easy to implement animations and to quickly provide look and feel.	More inclusive for customer, better look and feel for design
Drawbacks of prototyping tools	InVision is harder to use than Zeplin, but Zeplin lacks functionality.	None have been a complete solution. Have to use different ones in tandem. They don't provide usable code for developers.
Describe perfect prototyping tool	A combination of InVision and Zeplin. Provides a great deal of functionalities easily.	One tool to do everything.
First impression of MitMui	Good idea	Interesting concept, very fresh and pretty nice
Differences / opinions about MitMui tool and library		Yes, it depends how technical person you are and what you do
Benefits of new approach	Might be quicker than other tools, since the background work is already done	Seems very useful but also a bit limiting if only applicable to existing sites.
Drawbacks of new approach	Hard to say if it would in actuality be slower overall.	Not a all-in-one solution
Would you use	Yes, very interested	Yes, provided it was easy to use.

Figure C.2: Cystallized interview answers 2/3

Question / Interviewee	Interviewee 5, female (designer)	Interviewee 6, male (developer)
Development experience	0.5 years	several years, intensively 0.5 years
Design experience	officially 1 year, non professional 3 years	no professional
Interface prototyping experience	Yes, websites and posters	Yes, websites
Used prototyping tools and why	InVision, ease of use	InVision (suggested), Axure RP, Proto.io, Adobe XD
Good qualities of prototyping tools	Easy to showcase to customer	Quick, easy to provide look and feel, interactive, sharing
Drawbacks of prototyping tools	Customers might not realize the difference of protoype and actual product. Some things take actually longer to prototype than to just implement	Limiting, too much copying, online tools suffer from performance issues, platform dependency
Describe perfect prototyping tool	Combines design, prototyping and produces ready code for developers	Combination of Adobe XD and InVision Studio, interactive elements, ready assets, sharing,
First impression of MitMui	Wildly useful but mainly for pre-existing website/products. Doesn't seem useful if big changes are needed	Sounds reasonable and that it would work.
Differences / opinions about MitMui tool and library	I would use the graphical tool, don't know about development that much	The graphical tool has it uses, but not sure about the library.
Benefits of new approach	Seems great to be able to design and create a prototype at the same time.	Would lessen and quicken workload. Easy to create interactive prototypes based on existing content
Drawbacks of new approach	It might prove limiting having to work with pre-existing content. With blank canvas ideas might flow more freely	Prototypes created against certain version of site might get broken later. Provided html might not be usable for integration if modern frameworks are used
Would you use	Yes	Yes, possibly provided that it matched the requirements.

Figure C.3: Cystallized interview answers 3/3

Appendix D

Axure output HTML snippet

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page 1</title>
    <meta http-equiv="X-UA-Compatible" content="IE=edge"/>
    <meta http-equiv="content-type" content="text/html;
       charset=utf-8"/>
    <meta name="apple-mobile-web-app-capable" content="yes"/>
    <link href="resources/css/jquery-ui-themes.css" type="text/css"
       rel="stylesheet"/>
    <link href="resources/css/axure_rp_page.css" type="text/css"
       rel="stylesheet"/>
    <link href="data/styles.css" type="text/css" rel="stylesheet"/>
    <link href="files/page_1/styles.css" type="text/css"
       rel="stylesheet"/>
    <script src="resources/scripts/jquery-1.7.1.min.js"></script>
    <script
       src="resources/scripts/jquery-ui-1.8.10.custom.min.js"></script>
    <script src="resources/scripts/prototypePre.js"></script>
    <script src="data/document.js"></script>
    <script src="resources/scripts/prototypePost.js"></script>
    <script src="files/page_1/data.js"></script>
    <script type="text/javascript">
      $axure.utils.getTransparentGifPath = function() { return
         'resources/images/transparent.gif'; };
      $axure.utils.getOtherPath = function() { return
         'resources/Other.html'; };
      $axure.utils.getReloadPath = function() { return
         'resources/reload.html'; };
```

```
  </script>
</head>
<body>
  <div id="base" class="">

    <!-- viewmenu (Menu) -->
    <div id="u0" class="ax_default" data-label="viewmenu">
      <img id="u0_menu" class="img "
          src="images/page_1/viewmenu_u0_menu.png" alt="u0_menu"/>

      <!-- Unnamed (Table) -->
      <div id="u1" class="ax_default">

        <!-- Unnamed (Menu Item) -->
        <div id="u2" class="ax_default menu_item">
          <img id="u2_img" class="img "
              src="images/page_1/u2.png"/>
          <div id="u2_text" class="text ">
            <p><span>Item 1</span></p>
          </div>
        </div>

        <!-- Unnamed (Menu Item) -->
        <div id="u3" class="ax_default menu_item">
          <img id="u3_img" class="img "
              src="images/page_1/u2.png"/>
          <div id="u3_text" class="text ">
            <p><span>Item 2</span></p>
          </div>
        </div>

        <!-- Unnamed (Menu Item) -->
        <div id="u4" class="ax_default menu_item">
          <img id="u4_img" class="img "
              src="images/page_1/u4.png"/>
          <div id="u4_text" class="text ">
            <p><span>Item 3</span></p>
          </div>
        </div>
      </div>
    </div>

    <!-- Unnamed (Rectangle) -->
```

```
<div id="u5" class="ax_default paragraph">
  <div id="u5_div" class=""></div>
  <div id="u5_text" class="text ">
    <p><span>Lorem ipsum dolor sit amet, consectetur
        adipiscing elit. Aenean euismod bibendum laoreet.
        Proin gravida dolor sit amet lacus accumsan et viverra
        justo commodo. Proin sodales pulvinar sic tempor.
        Sociis natoque penatibus et magnis dis parturient
        montes, nascetur ridiculus mus. Nam fermentum, nulla
        luctus pharetra vulputate, felis tellus mollis orci,
        sed rhoncus pronin sapien nunc accuan eget.</span></p>
  </div>
</div>

<!-- Unnamed (Rectangle) -->
<div id="u6" class="ax_default heading_1">
  <div id="u6_div" class=""></div>
  <div id="u6_text" class="text ">
    <p><span style="font-family:'Arial Bold',
        'Arial';font-weight:700;">Heading 1</span></p>
  </div>
</div>

<!-- Unnamed (Rectangle) -->
<div id="u7" class="ax_default primary_button">
  <div id="u7_div" class=""></div>
  <div id="u7_text" class="text ">
    <p><span>BUTTON</span></p>
  </div>
</div>

<!-- Unnamed (Rectangle) -->
<div id="u8" class="ax_default button">
  <div id="u8_div" class=""></div>
  <div id="u8_text" class="text ">
    <p><span>BUTTON</span></p>
  </div>
</div>

<!-- Unnamed (Image) -->
<div id="u9" class="ax_default image ax_default_hidden"
    style="display:none; visibility: hidden">
  <img id="u9_img" class="img " src="images/page_1/u9.png"/>
```

```
</div>

<!-- Unnamed (Horizontal Line) -->
<div id="u10" class="ax_default line">
  <img id="u10_img" class="img " src="images/page_1/u10.png"/>
</div>

<!-- Unnamed (Rectangle) -->
<div id="u11" class="ax_default paragraph">
  <div id="u11_div" class=""></div>
  <div id="u11_text" class="text ">
    <p><span>Lorem ipsum dolor sit amet, consectetur
       adipiscing elit. Aenean euismod bibendum laoreet.
       Proin gravida dolor sit amet lacus accumsan et viverra
       justo commodo. Proin sodales pulvinar sic tempor.
       Sociis natoque penatibus et magnis dis parturient
       montes, nascetur ridiculus mus. Nam fermentum, nulla
       luctus pharetra vulputate, felis tellus mollis orci,
       sed rhoncus pronin sapien nunc accuan eget.</span></p>
  </div>
</div>

<!-- Unnamed (Table) -->
<div id="u12" class="ax_default">

  <!-- Unnamed (Table Cell) -->
  <div id="u13" class="ax_default table_cell">
    <img id="u13_img" class="img "
       src="images/page_1/u13.png"/>
    <div id="u13_text" class="text ">
      <p><span>Column 1</span></p>
    </div>
  </div>

  <!-- Unnamed (Table Cell) -->
  <div id="u14" class="ax_default table_cell">
    <img id="u14_img" class="img "
       src="images/page_1/u13.png"/>
    <div id="u14_text" class="text ">
      <p><span>Column 2</span></p>
    </div>
  </div>
```

```
    <!-- Unnamed (Table Cell) -->
    <div id="u15" class="ax_default table_cell">
      <img id="u15_img" class="img "
          src="images/page_1/u15.png"/>
      <div id="u15_text" class="text ">
        <p><span>Column 3</span></p>
      </div>
    </div>

    <!-- Unnamed (Table Cell) -->
    <div id="u16" class="ax_default table_cell">
      <img id="u16_img" class="img "
          src="images/page_1/u13.png"/>
    </div>

    <!-- Unnamed (Table Cell) -->
    <div id="u17" class="ax_default table_cell">
      <img id="u17_img" class="img "
          src="images/page_1/u13.png"/>
    </div>


  <!-- Unnamed (Menu) -->
  <div id="u22" class="ax_default">
    <img id="u22_menu" class="img "
        src="images/page_1/u22_menu.png" alt="u22_menu"/>

    <!-- Unnamed (Table) -->
    <div id="u23" class="ax_default">

      <!-- Unnamed (Menu Item) -->
      <div id="u24" class="ax_default menu_item">
        <img id="u24_img" class="img "
            src="images/page_1/u24.png"/>
        <div id="u24_text" class="text ">
          <p><span>File</span></p>
        </div>
      </div>

      <!-- Unnamed (Menu Item) -->
      <div id="u25" class="ax_default menu_item">
        <img id="u25_img" class="img "
            src="images/page_1/u24.png"/>
```

```
            <div id="u25_text" class="text ">
              <p><span>Edit</span></p>
            </div>
          </div>

          <!-- Unnamed (Menu Item) -->
          <div id="u26" class="ax_default menu_item">
            <img id="u26_img" class="img "
                src="images/page_1/u26.png"/>
            <div id="u26_text" class="text ">
              <p><span>View</span></p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```